НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
Кафедра автоматизованих систем обробки інформації та управління

УДК 519.85

«До захисту допущено»
В.о. завідувача кафедри
_____ *О.А.Павлов*
(підпис) (ініціали, прізвище)
“___”_____2019 р.

Дипломний проект
на здобуття ступеня бакалавра

з напряму підготовки *6.050101 «Комп’ютерні науки»*

на тему: «Інформаційна система підтримки розвитку та самоорганізації особистості»

Виконав: студент *4* курсу, групи *ІС-52*

Черков Павло Юрійович _____
(прізвище, ім’я, по батькові) (підпис)

Керівник *доц., к.т.н. Сперкач М.О.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант з графічної документації *ст.викл. Халус О.А.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент *доц., к.т.н., доц. Писаренко А.В.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____
(підпис)

Київ – 2019 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет (інститут) *інформатики та обчислювальної техніки*
(повна назва)

Кафедра *автоматизованих систем обробки інформації та управління*
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки (програма професійного спрямування) *6.050101*

«Комп’ютерні науки» («Інформаційні управляючі системи та технології»)

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ *О.А. Павлов*
(підпис) (ініціали, прізвище)
“___”_____2019 р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ СТУДЕНТУ

Черкова Павла Юрійовича
(прізвище, ім’я, по батькові)

1. Тема проекту «Інформаційна система підтримки розвитку та самоорганізації особистості»

керівник проекту *Сперкач Майя Олегівна, к.т.н.*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від “*23*”*квітня* 2019 р. №1181-с

2. Термін подання студентом проекту “*03*”*червня* 2019 року

3. Вихідні дані до проекту

Технічне завдання

4. Зміст пояснювальної записки

1. Загальні положення: основні визначення та терміни, опис предметного середовища, огляд ринку програмних продуктів, постановка задачі

2. Інформаційне забезпечення: вхідні дані, вихідні дані, опис структури бази даних

3. Математичне забезпечення: змістовна та математична постановки задачі, обґрунтування та опис методу розв’язання

4. Програмне та технічне забезпечення: засоби розробки, вимоги до технічного забезпечення, архітектура програмного забезпечення

5. Технологічний розділ: керівництво користувача, методика випробувань програмного продукту

5. Перелік графічного матеріалу

1. Схема структурна варіантів використань

2. Схема бази даних

3. Схема структурна діяльності

4. Схема структурна взаємодії компонентів

5. Схема структурна послідовності

6. Креслення вигляду екранних форм

6. Консультанти розділів проекту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання *«23» квітня 2019 року*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів проекту	Примітка
1.	*Вивчення рекомендованої літератури*	*16.02.2019*	
2.	*Аналіз існуючих методів розв'язання задачі*	*23.02.2019*	
3.	*Постановка та формалізація задачі*	*01.03.2019*	
4.	*Розробка інформаційного забезпечення*	*11.03.2019*	
5.	*Алгоритмізація задачі*	*17.03.2019*	
6.	*Обґрунтування використовуваних технічних засобів*	*23.03.2019*	
7.	*Розробка програмного забезпечення*	*28.03.2019*	
8.	*Налагодження програми*	*03.04.2019*	
9.	*Виконання графічних документів*	*11.04.2019*	
10.	*Оформлення пояснювальної записки*	*23.04.2019*	
11.	*Подання ДП на попередній захист*	*30.05.2019*	
12.	*Подання ДП на основний захист*	*03.06.2019*	
13.	*Подання ДП рецензенту*	*05.06.2019*	

Студент _____ П.Ю. Черков
(підпис)

Керівник проекту _____ М.О. Сперкач
(підпис)

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
			Документи текстові		
1	А4	ДП ІС-5224.1181-с.ПЗ	Пояснювальна записка	65	
2	А4	ДП ІС-5224.1181-с.ТЗ	Технічне завдання	10	
3	А4	ДП ІС-5224.1181-с.ПМВ	Програма та методика випробувань	9	
			Матеріали графічні		
4	А3	ДП ІС-5224.1181-с.ССВ	Схема структурна варіантів використань	1	
5	А4	ДП ІС-5224.1181-с.СБД	Схема бази даних	1	
6	А4	ДП ІС-5224.1181-с.ССД	Схема структурна діяльності	3	
7	А4	ДП ІС-5224.1181-с.ССК	Схема структурна взаємодії компонентів	1	
8	А4	ДП ІС-5224.1181-с.ССП	Схема структурна послідовності	1	
9	А3	ДП ІС-5224.1181-с.КЕ	Креслення вигляду екранних форм	1	

Зм.	Арк.	ПІП	Підп.	Дата	ДП ІС-5224.1181-с.ВДП		
Розробн.		Черков П.Ю.			Відомість дипломного проекту	Літ.	Лист 1 / Листів 1
Керівн.		Сперкач М.О.					
Консульт.						КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52	
Н/контр.		Халус. О.А.					
В.о.зав.каф.		Павлов О.А.					

Пояснювальна записка
до дипломного проекту

на тему: Інформаційна система підтримки розвитку та самоорганізації особистості

Київ – 2019 року

АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка дипломного проекту складається з п’яти розділів, містить 22 рисунків, 20 таблиць, 1 додаток, 15 джерел.

Дипломний проект присвячений розробці системи для підвищення рівня самоорганізації особистості та її розвитку. Процес покращення стану життя спеціалістами є завданням з великим шансом на невдалий результат, у зв’язку з необхідністю власноруч формулювати план для досягнення цілей

У розділі з інформаційного забезпечення були визначені дані для реєстрації користувача в системі, дані для опису завдань, вхідні та вихідні дані до кожної сфери життя, були розроблені вимоги до структури вхідних даних, що полегшують реалізацію поставлених комплексних задач.

Розділ математичного забезпечення присвячений обґрунтуванню обраного підходу перестановки та формулювання розкладу, що дозволить зменшити загальний час, необхідний на реалізацію поставлених завдань.

Розділ програмного забезпечення описує основні засоби розробки системи, висунуті вимоги до технічного забезпечення. В цьому розділі обрано та обґрунтовано архітектуру програмного забезпечення.

У технологічному розділі описана інструкція користувача та проведене тестування системи.

ТЕОРІЯ РОЗКЛАДУ, КОЛЕСО ЖИТТЯ, САМОРОЗВИТОК, ДОСЯГНЕННЯ ЦІЛЕЙ, ТАЙМ-МЕНЕДЖМЕНТ, КАЛЕНДАРНИЙ ПЛАН.

ABSTRACT

Structure and scope of work. Diploma project consists of six sections, contains 22 drawings, 20 tables, 1 applications, 15 sources.

The diploma project is devoted to the development of a system for increasing the level of self-organization of the individual and its development. The process of improving the quality of life by professionals is a challenge with a great chance for an unsuccessful outcome, due to the need to formulate a plan for achieving its goals.

In the information provision section, the data for user registration in the system were defined, data for the description of tasks, input and output data for each sphere of life, requirements for the structure of the input data facilitating the implementation of the set of complex tasks were developed.

The section of mathematical provision is devoted to substantiation of the chosen approach of permutation and formulation of the schedule, which will reduce the total time necessary for the implementation of the tasks.

The software section describes the main tools of the system development, the requirements for technical support. This section defines and justifies the software architecture.

The technological section describes the user's manual and tests the system.

THEORY OF THE SCHEDULE, WHEEL LIFE, SELF-TRANSPARENCY, AIMS AIM, TIME-MANAGEMENT, CALENDAR PLAN.

ЗМІСТ

ВСТУП **5**

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ **7**

1.1 Опис предметного середовища 7

1.1.1 Опис процесу діяльності *8*

1.1.2 Опис функціональної моделі *9*

1.2 Огляд наявних аналогів 10

1.3 Постановка задачі 14

1.3.1 Призначення розробки *14*

1.3.2 Цілі та задачі розробки *14*

Висновок до розділу 15

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ **16**

2.1 Вхідні дані 16

2.2 Вихідні дані 17

2.3 Опис структури бази даних 19

Висновок до розділу 25

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ **26**

3.1 Змістовна постановка задачі 26

3.2 Математична постановка задачі 26

3.3 Обґрунтування методу розв’язання 27

3.4 Опис методів розв’язання 28

Висновок до розділу 30

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ **31**

4.1 Засоби розробки 31

4.2 Вимоги до технічного забезпечення 33

4.3 Архітектура програмного забезпечення 33

4.4 Специфікація функцій 36

Висновок до розділу 39

5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ **40**

5.1 Керівництво користувача 40
5.2 Випробування програмного продукту 49
5.2.1 Мета випробувань 49
5.2.2 Загальні положення 49
5.2.3 Результати випробувань 50
Висновок до розділу 52
ЗАГАЛЬНІ ВИСНОВКИ 55
ПЕРЕЛІК ПОСИЛАНЬ 57
ЗАСТОСУНОК А 59

ВСТУП

У двадцять першому столітті час пливе неймовірно швидко. Усі постійно кудись поспішають. Багато людей, які поглинуті у роботу, або зайняті особистими речами дуже часто забувають про те, що людина має розвиватися в усіх напрямках. Адже окрім кар'єри, або спорту є й інші речі, які необхідні для повноцінного насолодження життям.

Ми часто ходимо по колу. Нам дуже важко побачити, що відбувається у нашому житі і як ми можемо вплинути на розвиток подій. У кругообігу часу, щоденній метушні та безкінечного списку справ, зупинитися, для того, щоб опрокинути поглядом своє буття, досить складна задача. Та досить відкладати свій прогрес на майбутнє. Настала пора побачити своє життя з «висоти пташиного польоту».

Саме для цього придумали дуже ефективне тренування під назвою «Колесо життєвого балансу» або «Колесо життя», яке дає змогу зрозуміти:

— з чого складається ваше життя;
— що саме для вас дійсно важливо;
— зміни в якій області викличуть кардинальні покращення у вашому існуванні;
— які кроки ви маєте змогу зробити, щоб вже за 72 години поліпшити свій побут.

Цю техніку винайшов доктор психологічних наук, та автор 24 книг по психології Пол Мейер. На сьогоднішній день, даний спосіб використовують, на багатьох тренінгах у різних інтерпретаціях. Найкращий опис використання «Колеса життя» представлено у книзі «Екстримальний тайм-менеджмент».

Кожен мріє про те, щоб мати змогу проявити себе, заробляти багато грошей, купити дім, завести сім'ю, зайняти необхідне місце в цьому житті. Та мрії залишаються мріями, доти, доки не будуть десь задокументовані.

Тільки після цього мрії стають цілями. Використання веб-технологій для структуризації справ або цілей – є дуже популярним рішенням на сьогоднішній день.

Метою проекту є автоматизація техніки «Колесо життя» та складання оптимізованого розкладу задач для досягнення поставлених цілей, що сприяє кращому та швидшому розвитку людини, як особистості та надає змогу завжди мати список справ при собі.

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Опис предметного середовища

Серед успішних та організованих людей актуальною є тема розвитку та самоорганізації особистості. Нині дуже популярно вести здоровий спосіб життя та слідкувати за розпорядком дня, адже це є ключем до успіху. Для цього винайшли багато різних засобів, що допомагають у структуризації щоденних планів.

У данному предметному середовищі буде розглядатися інформаційна система у вигляді онлайн-щоденника.

Онлайн-щоденник – це технологія, яка потрібна всім сучасним та успішним людям, які хочуть розкласти всі свої справи по полицям задля кращого виконання планів.

Існує така методика аналізу життєвого балансу як «Колесо життя» [1]. Відомо, що життєвий баланс – це не просто комфортне виживання, а повноцінне цікаве та активне життя. Щоб продуктивно та захопливо жити, необхідно як мінімум одне-два заняття, які приносять радість. Багато людей не вміють керувати своїм часом, а тому шлях до балансу стає нелегким. Так, для покращення взаємодії з оточенням та для покращення рівня життя кожній людині необхідно мати статистику її справ.

Для досягнення балансу в житті необхідно тримати в порядку всі сектори життя. «Колесо» розбито на 8 секторів: здоров’я, кар’єра, оточення, сім’я та стосунки, відпочинок, особистий ріст, творчість, духовність. Кожну сферу необхідно оцінити від 1 до 10 балів:

– 1 – усе погано,

– 10 – усе неперевершено.

У результаті виходить колесо життєвого балансу (рисунок 1.1).

Рисунок 1.1 – Колесо балансу життя

Сфера життя, що набрала мало балів – головне джерело незадоволення в житті. Покращуючи її, зросте й відчуття загального благополуччя.

Оцінювати усі сфери необхідно раз у 2-3 тижні, щоб відслідковувати прогрес та ставати більш успішним.

У данній роботі буде розглянуто розробку онлайн-щоденника, що матиме змогу оптимізувати розклад справ на кожен день, а також генерувати статистику по результатам та оцінкам виконаних робіт.

1.1.1 Опис процесу діяльності

Опис процесу діяльності представлено у вигляді трьох незалежних процесах.

Перший процес – формування діаграми візуалізації теперішнього рівня життя, відображений у графічному матеріалі.

Другий процес – формування списку справ на спринт, що відображений у графічному матеріалі.

Третій процес – візуалізація графіку покращення та досягнення найкращого результату в певній сфері, що відображений у графічному матеріалі.

1.1.2 Опис функціональної моделі

Опишемо функціональну модель системи. Актором системи є клієнт. Основними функціями системи є:

- формування статистики;
- формування діаграми життя;
- формування розкладу;
- формування плану спринта.

Візуалізуємо функціональну модель у вигляді схеми структурної варіантів використання, яка відображена у графічному матеріалі.

Розглянемо детальніше функціональні вимоги (таблиця 1.1).

Таблиця 1.1 – Функціональні вимоги

Варіант використання	**Функціональна вимога**	**Пріоритет**
Формування діаграми життя	1. Система надає можливість сформувати діаграму життя згідно з оцінкою кожної сфери	Високий
Вибір сфери	1.1. Система надає можливість обрати певну сферу життя	Високий
Введення цілі на спринт	1.2. Система надає можливість ввести ціль на спринт для обраної сфери	Високий
Введення завдань для цілі на спринт	1.3. Система надає можливість ввести завдання для обраної цілі спринт	Середній
Оцінка поточного стану сфери життя	1.4. Система надає можливість користувачеві оцінити поточний стан сфери за шкалою від 1 до 10 (1 – усе погано, 10 – усе неперевершено)	Високий

Продовження таблиці 1.1

Формування статистики	2. Система надає можливість сформувати статистику згідно з даними, заповненими користувачем	Високий
Вибір сфери життя	2.1. Система надає можливість обрати сферу життя (здоров’я, кар’єра, оточення, сім’я та стосунки, відпочинок, особистий ріст, творчість, духовність) для формування статистики	Високий
Формування та відображення статистики	2.2. Система надає можливість сформуватиі та відобразити статистику за обраною сферою життя (здоров’я, кар’єра, оточення, сім’я та стосунки, відпочинок, особистий ріст, творчість, духовність)	Середній
Формування розкладу	3. Система надає можливість сформувати розклад за допомогою функції покращення раціональності розкладу	Середній
Записати справу	3.1. Система надає можливість записати справу на день	Високий
Помітити можливість зміни в розкладі	3.1.1. Система надає можливість помітити справу, як справу з жорстко встановленним часом	Середній
Обрати час виконання	3.1.2. Система надає можливість обрати проміжок часу для виконання справ	Середній
Формування плану спринта	4. Система надає можливість сформувати план спринта	Високий
Відмітка про виконання справи	4.1. Система надає можливіть поставити відмітку того, що справу виконано	Середній

1.2 Огляд наявних аналогів

У таблиці 1.2 наведено наявні аналоги запропонованого програмного забезпечення.

Таблиця 1.2 – Наявні аналоги

Назва	Режим доступу
Google календар	https://calendar.google.com/calendar/r
TickTick	https://ticktick.com/
AnyDo	https://web.any.do/
World Time Buddy	https://www.worldtimebuddy.com/
Hours Time Tracking	https://www.hourstimetracking.com/

Розглянемо детальніше кожен із наведених аналогів.

Google календар

Google календар – це безкоштовий веб-застосунок, що призначений для тайм-менеджменту та розроблений компанією Google [2].

Мова написання – Java.

Даний аналог працює на підході до побудови користувацьких веб-застосунків Ajax. Це надає змогу користувачам переглядати, додавати та перетягувати події з однієї дати на іншу не перезавантажуючи сторінку. Користувач має змогу змогу переглядати плани не лише на конкретний день, а й на тиждень та місяць [3].

Google календар можна завантажити на Android чи iOs як окремий дотаток, а також ним можна користуватися зайшовши із будь-якого браузеру. Важливим є те, що всі події зберігаються онлайн, а також можна додавати, та обмінюватись багатьма календарями з різними рівнями прав доступу [4].

TickTick

TickTick – це безкоштовний застосунок, що дозволяє користувачам записувати список справ на кожен день. Він може виступати органайзером та планувальником.

TickTick позиціонує себе як трекер звичок. Користувач може створювати списки, а саме:

– перший рівень – списки, що розглядаються як набір проектів;

– другий рівень – задачі в обраному списку;

– третій рівень – чек-листи в рамках обраної задачі.

Даний аналог дає змогу синхронізувати дані між усіма пристроями. Тут користувач може відмічати виконання поставлених задач, додавати нові задачі та переключатися між списками [5].

Наведемо основні функції, якими володіє TickTick: голосовий набір тексту; кастомізований лист задач, підзадач, чек-лист, що можна об'єднувати, фільтрувати та об’єднувати в тематичні папки; перегляд своєї статистики продуктивності [6].

AnyDo

AnyDo – це безкоштовний застосунок, що призначений для планування часу, написання списку справ та нагадувань. Застосунок дає змогу відслідковувати прогрес. AnyDo надає можливість автоматично синхронізувати задачі на всіх пристроях у режимі реального часу [7].

Застосунок надає можливість синхронізувати дані з календарем телефону, календарем Google, подіями на Facebook, надає можливість голосового вводу справ, написання заміток. Можна також здійснювати написання списку покупок. Важливою функцією є проста організація проектів, а саме: можливість ділитися списком справ з іншими користувачами [8].

World Time Buddy

World Time Buddy – це платний англомовний застосунок, що дозволяє планувати. Головною функцією цього застосунку є синхронізація різних часових поясів. Це є зручним для людей, що подорожують по планеті та часто змінюють свої часові пояси.

Список справ користувач може записувати на конкретну дату. Усі справи відображені в календарі, при натисненні на конкретну дату, можна переглянути заплановані справи. застосунок також посилає нагадування, що дозволяє користувачеві не забувати про заплановану задачу [9].

Hours Time Tracking

Hours Time Tracking – це платний англомовний застосунок, що призначений для планування.

Цей застосунок має візуальну шкалу часу, розумні нагадування. Користувач має змогу обрати Професійний, командний чи Персональний режим роботи залежно від того, які функції необхідні. Користувач може створювати таймери, що дозволяють відслідковувати виконання задач у режимі реального часу. Ці таймери користувач має змогу перемикати у разі виконання декількох задач одночасно. Візуальну шкалу часу користувач має змогу редагувати, а також у застосунку присутня синхронізація між усіма пристроями [10].

Проаналізуємо знайдені аналоги на наявність у них потрібного функціоналу (таблиця 1.3).

Таблиця 1.3 – Огляд функцій наявних аналогів

Аналоги / **Функції**	**Google календар**	**TickTick**	**AnyDo**	**World Time Buddy**	**Hours Time Tracking**
Візуалізація діаграми життя	–	–	–	–	–
Формування розпорядку дня	+	+	+	+	+
Оптимізація розпорядку дня	–	–	–	–	–
Візуалізація статистики та графіків відповідної сфери	+	–	–	+	+

Отже, як бачимо з таблиці 1.3, усі ці системи не задовольняють виконанню всіх функцій, які було використано при розробці нашої інформаційної системи.

1.3 Постановка задачі

1.3.1 Призначення розробки

Система призначена для підтримки розвитку та самоорганізації особистості.

1.3.2 Мета та задачі розробки

Метою розробки є підвищення рівня самоорганізації особистості та її розвитку.

Для досягнення поставленої мети мають бути вирішені такі задачі:

- генерація діаграми розвитку сфер життя;
- створення кінцевих цілей для кожної сфери;
- формування плану спринта;
- формування статистики.

Висновок до розділу

У розділі з описом предметного середовища було описано процес діяльності. У данному предметному середовищі буде розглядатися інформаційна система у вигляді онлайн-щоденника. Існує така методика аналізу життєвого балансу як «Колесо життя».

Багато людей не вміють керувати своїм часом, а тому шлях до балансу стає нелегким. Так, для покращення взаємодії з оточенням та для покращення рівня життя кожній людині необхідно мати статистику її справ.

Для досягнення балансу в житті необхідно тримати в порядку всі сектори життя. «Колесо» розбито на 8 секторів: здоров’я, кар’єра, оточення, сім’я та стосунки, відпочинок, особистий ріст, творчість, духовність. Кожну сферу необхідно оцінити від 1 до 10 балів:

– 1 – усе погано,

– 10 – усе неперевершено.

У данній роботі буде розглянуто розробку онлайн-щоденника, що матиме змогу оптимізувати розклад справ на кожен день, а також генерувати статистику по результатам та оцінкам виконаних робіт.

Також було описано фнкціональну модель, визначено основного користувача системи – клієнта, проаналізовано його основні функції.

Для визначення переваг системи, що реалізована в дипломному проекті, було розглянуто наявні аналоги.

У одному з підрозділів було визначено постановку задачі, а саме: призначення розробки, мету та задачі розробки.

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Вхідні дані

Перед початком користування програмою користувач має зареєструватися. Дані по кожному користувачу зберігаються в окремій колекції. Ця колекція містить окремий документ по користувачам.

Атрибутами колекції є:

– поштовий адрес користувача;

– пароль користувача.

У разі, якщо користувач зареєстрований, то він проходить авторизацію. У цьому випадку дані підтягуються з бази даних.

Для системи, що розробляється в дипломному проекті, на вході отримано вісім документів для кожної сфери, що містять у собі інформацію про кожну сферу. Кожна сфера включає в себе:

– цілі;

– оцінки;

– завдання.

Вхідні документи збережені у форматі .json.

Атрибутами сфери є:

– sphereName: назва сфери;

– sphereId: унікальний номер сфери;

– sphereAims: масив списку цілей для сфери;

– marks: масив поточних оцінок сфери;

– tasks: масив поточних завдань сфери.

Кожен масив включає в себе відповідну інформацію. Розглянемо детальніше кожен масив.

– sphereAims. Атрибути:

1) aimName: назва цілі;

2) estimatedDate: визначена дата для завершення виконання цілі;

3) aimId: унікальний номер цілі;

4) subtasks: масив підцілей для цілі, що, у свою чергу містить у собі наступні атрибути:

a. subAimName: назва підцілі;

b. estimatedDate: визначена дата для завершення виконання підцілі.

- marks. Атрибути:
  1) todayMark: поточна оцінка цілі;
  2) date: дата виставлення поточної оцінки цілі.
- tasks. Атрибути:
  1) taskName: назва завдання сфери;
  2) taskDate: дата виконання завдання;
  3) taskTime: час виконання завдання;
  4) agileMark: помітка гнучкості;
  5) done: помітка користувача про виконання справи.

2.2 Вихідні дані

Вихідні дані використовуються для відображення:

- діаграми життя;
- статистики рівня розвитку користувача поденно для кожного спринта по певній сфері;
- календаря зі списком справ.

Розглянемо детальніше кожний тип вихідних даних.

На рисунку 2.1 відображено діаграму життя.

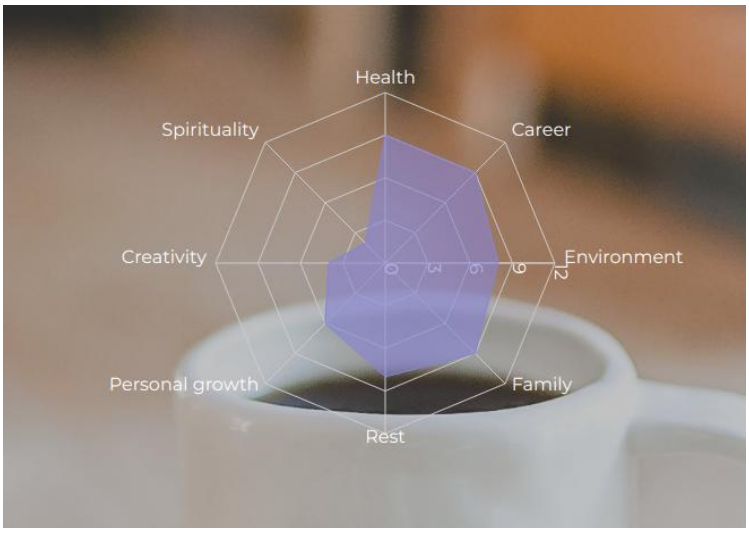


Рисунок 2.1 – Діаграма життя

Бачимо, що на діаграмі життя відображено поточні оцінки користувача для кожної сфери.

На рисунку 2.2 зображено статистистичну інформацію рівня розвитку користувача щотижнево для спринта по певній сфері.

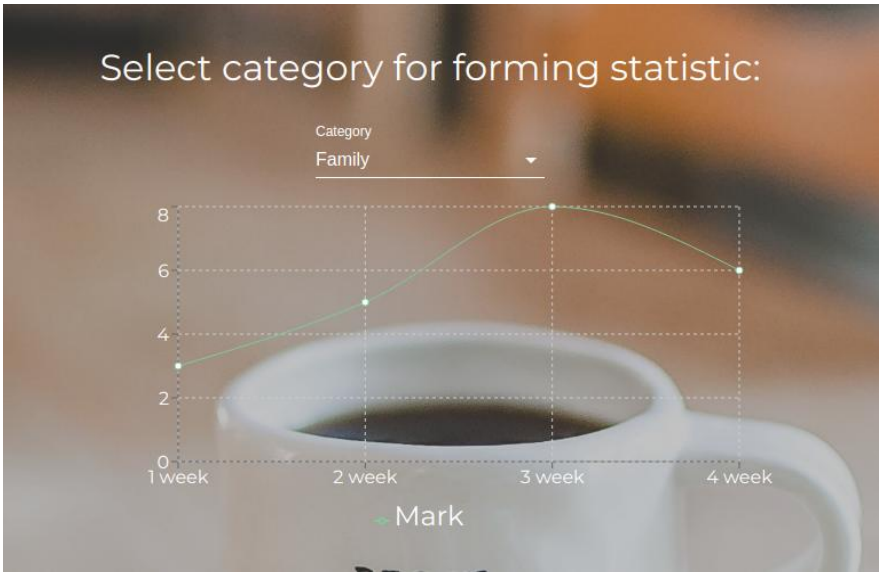


Рисунок 2.2 – Діаграма статистики розвитку певної сфери життя

Бачимо, що на діаграмі статистики розвитку певної сфери життя відображено дані по кожній сфері. По осі ОХ відображено дати, за які відображено інформацію, по осі ОУ відображено оцінки для кожної сфери.

Таким чином користувач може відслідковувати прогрес чи регрес своєї діяльності.

На рисунку 2.3 наведено календар зі списком завдань.

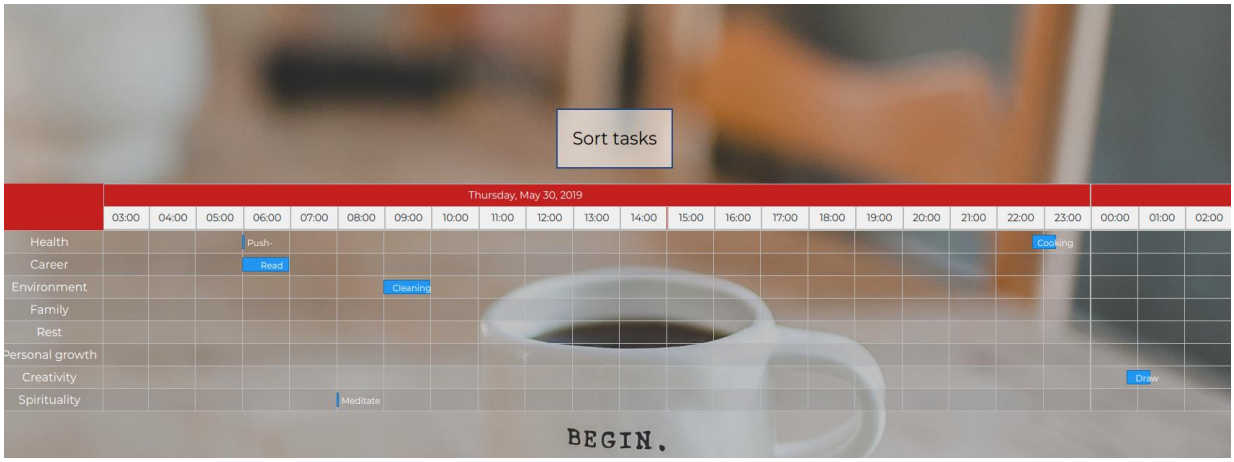


Рисунок 2.3 – Screen Shot календаря зі списком справ

Бачимо, що на Screen Shot-і календаря зі списком справ відображено спарави користувача на обраний день. Тут користувач може додавати справи, редагувати та видаляти, а також ставити відмітку «виконано», якщо справу було зроблено.

2.3 Опис структури бази даних

Для нашого програмного продукту було використано БД MongoDB. Тут міститься дві колекції:

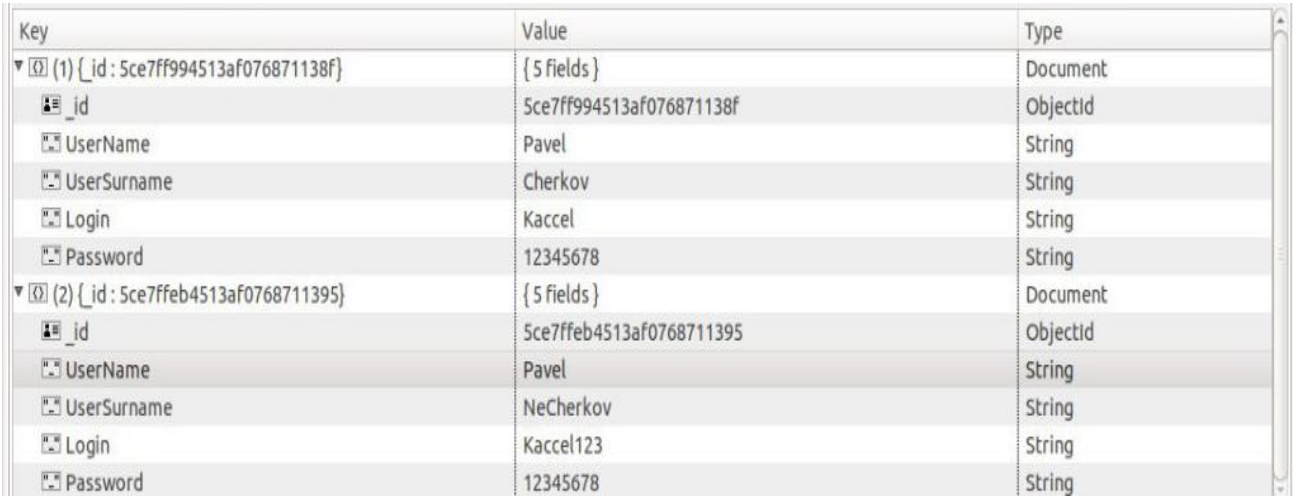
– Users;

– Categories.

Структурна схема БД наведена у графічних матеріалах у відповідному розділі

Розглянемо детальніше кожну колекцію БД.

На рисунку 2.4 наведено Screen Shot структури колекції Users.



Key	Value	Type
(1) { _id : 5ce7ff994513af076871138f}	{5 fields}	Document
_id	5ce7ff994513af076871138f	ObjectId
UserName	Pavel	String
UserSurname	Cherkov	String
Login	Kaccel	String
Password	12345678	String
(2) { _id : 5ce7ffeb4513af0768711395}	{5 fields}	Document
_id	5ce7ffeb4513af0768711395	ObjectId
UserName	Pavel	String
UserSurname	NeCherkov	String
Login	Kaccel123	String
Password	12345678	String

Рисунок 2.4 – Структура колекції Users

Атрибутами документу User в колекції Users є:

– id: це унікальний ідентифікатор документу, що генерується самою БД MongoDB;

– UserName: ім’я користувача, введене при реєстрації;

– UserSurname: прізвище користувача, введене користувачем при реєстрації;

– Login: це логін, введений при реєстрації;

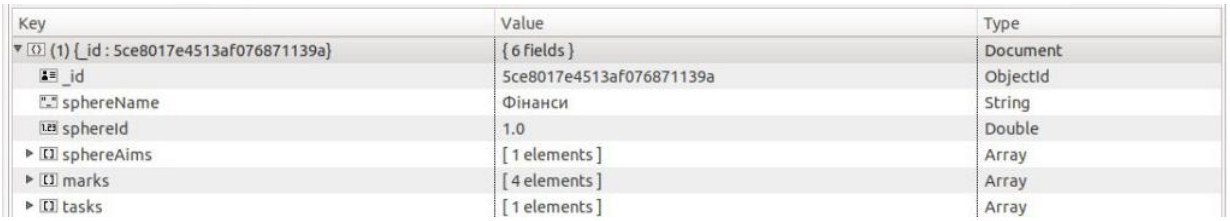
– Password: це пароль, введений користувачем при реєстрації.

У таблиці 2.1 наведено типи даних для кожного атрибуту колекції Users.

Таблиця 2.1 – Типи даних атрибутів колекції Users

Атрибут	**Тип**
id	ObjectId
UserName	String
UserSurname	String
Login	String
Password	String

На рисунку 2.5 наведено Screen Shot структури колекції Categories.



Key	Value	Type
(1) {_id : 5ce8017e4513af076871139a}	{6 fields}	Document
_id	5ce8017e4513af076871139a	ObjectId
sphereName	Фінанси	String
sphereId	1.0	Double
sphereAims	[1 elements]	Array
marks	[4 elements]	Array
tasks	[1 elements]	Array

Рисунок 2.5 – Структура колекції Categories

Атрибутами документу Categories в колекції Categories є:

– id: це унікальний ідентифікатор документу, що генерується самою БД MongoDB;

– sphereName: назва сфери;

– sphereId: унікальний номер сфери;

– sphereAims: масив списку цілей для сфери;

– marks: масив поточних оцінок сфери;

– tasks: масив поточних завдань сфери.

У таблиці 2.2 наведено типи даних для кожного атрибуту документу Categories в колекції Categories.

Таблиця 2.2 – Типи даних атрибутів документу Categories в колекції Categories

Атрибут	**Тип**
id	ObjectId
sphereName	String
sphereId	Double
sphereAims	Array
marks	Array
tasks	Array

На рисунку 2.6 наведено Screen Shot структури масиву списку цілей для сфери sphereAims.

Рисунок 2.6 – Структура масиву списку цілей для сфери sphereAims

Атрибутами структури масиву списку цілей для сфери sphereAims є:

– aimName: назва цілі;

– estimatedDate: визначена дата для завершення виконання цілі;

– aimId: унікальний номер цілі;

– subtasks: масив підцілей для цілі, що, у свою чергу містить у собі наступні атрибути.

У таблиці 2.3 наведено типи даних для кожного атрибуту структури масиву списку цілей для сфери sphereAims.

Таблиця 2.3 – Типи даних атрибутів структури масиву списку цілей для сфери sphereAims

Атрибут	**Тип**
aimName	String
estimatedDate	String
aimId	Double
subtasks	Array

Для масиву subtasks є свої атрибути, а саме:

– subAimName: назва підцілі;

– estimatedDate: визначена дата для завершення виконання підцілі.

У таблиці 2.4 наведено типи даних для атрибутів масиву subtasks.

Таблиця 2.4 – Типи даних атрибутів масиву subtasks

Атрибут	**Тип**
subAimName	String
estimatedDate	String

На рисунку 2.7 наведено Screen Shot структури масиву поточних оцінок сфери marks.

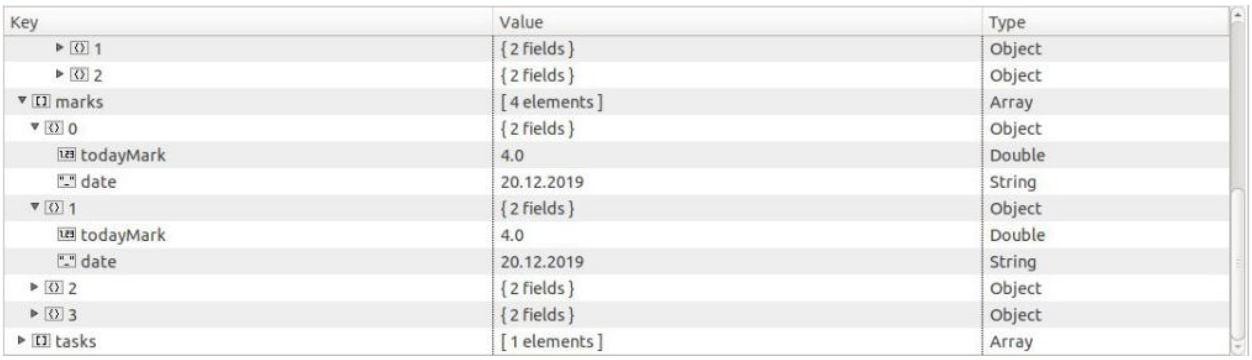


Рисунок 2.7 – Структура масиву поточних оцінок сфери marks

Атрибутами структури масиву поточних оцінок сфери marks є:

– todayMark: поточна оцінка цілі;

– date: дата виставлення поточної оцінки цілі.

У таблиці 2.5 наведено типи даних для атрибутів масиву поточних оцінок сфери marks.

Таблиця 2.5 – Типи даних атрибутів масиву поточних оцінок сфери marks

Атрибут	**Тип**
todayMark	Double
date	String

На рисунку 2.8 наведено Screen Shot структури масиву поточних завдань сфери tasks.

Рисунок 2.8 – Структура масиву поточних завдань сфери tasks

Атрибутами структури масиву поточних завдань сфери tasks є:

– taskName: назва завдання сфери;

– taskDate: дата виконання завдання;

– taskTime: час виконання завдання;

– agileMark: помітка гнучкості;

– done: помітка користувача про виконання справи.

У таблиці 2.6 наведено типи даних для кожного атрибуту масиву поточних завдань сфери tasks.

Таблиця 2.6 – Типи даних атрибутів масиву поточних завдань сфери tasks

Атрибут	**Тип**
taskName	String
taskDate	String
taskTime	String
agileMark	String
done	Bool

Висновок до розділу

У розділі інформаційного забезпечення було описано вхідні та вихідні дані.

Визначено, що вхідними даними до системи є колекції Users та Categories. У колекції Users зберігаються дані, що необхідні для ідентифікації клієнтів у системах. У колекції Categories зберігається детальна інформаццція, що вводить користувач по кожній категорії.

Вихідними даними системи є:

– діаграми життя;

– статистики рівня розвитку користувача поденно для кожного спринта по певній сфері;

– календаря зі списком справ.

Одним із підрозділів є опис структури бази даних. Для нашого програмного продукту було використано БД MongoDB. У даному підрозділі описано структуру зберігання даних у БД та представлена схема БД. Було наведено опис кожної колекції та її атрибутів, а також опис кожного масиву з його атрибутами.

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Змістовна постановка задачі

Нехай задано список завдань, кожне з яких містить час на виконання, термін виконання, опис завдання та залежність від інших завдань. Необхідно перерозподілити план виконання усіх завдань, так щоб мінімізувати час виконання усіх завдань.

3.2 Математична постановка задачі

Нехай

- m - 1;
- множину орграфів робіт $\overline{G} = \{G_1, G_2, ..., G_n\}$, де G_i - орграф залежності операцій роботи i.

Роботи мають такі характеристики:

- n - кількість робіт;
- r_i - момент готовності i, $i = \overline{1,n}$;
- d_i - плановий (директивний) термін;
- $a_i = d_i - r_i$ - допустима тривалість проходження роботи в системі;
- u_i - вага (пріоритет) i-ої роботи, $i = \overline{1,n}$;
- g_i - кількість операцій i-ої роботи, $i = \overline{1,n}$.

Для j-ої операції і-ої роботи $(i = \overline{1,n},\ j = \overline{1,g_i})$ задаються:

- m_{ij} - номер машини, на якій виконується ця операція, $1 \le m_{ij} \le m$;
- t_{ij} - тривалість виконання операції;
- $t_i = \sum_{j=1}^{g_i} t_{ij}$ - загальна тривалість всіх операцій роботи i (тривалість роботи).

На операції роботи накладено відношення передування.

Позначимо через W_{ij} − час очікування операції ij, тобто інтервал часу між закінченням $(j-1)$-ої операції і початком j-ої операції i-ої роботи. Тоді загальна тривалість очікування роботи дорівнює сумі тривалостей очікування всіх операцій:

$$W_i = \sum_{j=1}^{g_i} W_{ij} .$$

Необхідно впорядкувати список робіт так, щоб мінімізувати сумарний зваженний момент закінчення

$$T_i = r_i + W_{i1} + t_{i1} + W_{i2} + t_{i2} + ... + W_{ig_i} + t_{ig_i} =$$

$$= r_i + \sum_{j=1}^{g_i} t_{ij} + \sum_{j=1}^{g_i} W_{ij} = r_i + t_i + W_i, \quad ;$$

$$\sum_{i=1}^{n} T_i \to \min .$$

3.3 Обґрунтування методу розв'язання

В теорії розкладів розглядаються задачі впорядкування при умові, що розв'язані всі питання, які відносяться до того, що і яким чином повинно бути виконано. При цьому припускається, що не існує залежності між характером цих розв'язків і встановлюється порядком, тобто, характер робіт не залежить від їх послідовності виконання [11]. «Характер робіт залежить від їх послідовності виконання» означає, що тривалість виконання будь-якої роботи (робіт) залежить від її місця в розкладі. наприклад:

- від того яка (які) роботи знаходяться в розкладі перед нею або після неї;
- від того, чи виконана чи ні на поточний момент деяка (деякі) робота;

– від того, в якому порядку виконувалися попередні роботи; від моменту початку роботи і т. п.

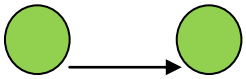
3.4 Опис методів розв'язання

Нехай маємо систему, в якій $m=2$, $n=3$.

Робота A ($g_A=3$):

$m_{A1}=1, m_{A2}=2, m_{A3}=1$, $t_{A1}=3, t_{A2}=2, t_{A3}=7$.

Робота B ($g_B=2$):



$m_{B1}=1, m_{B2}=2$, $t_{B1}=5, t_{B2}=5$.

Робота C ($g_C=1$):

$m_{C1}=2$, $t_{C1}=4$.

І нехай $d_A=d_B=d_C=d$.

На рисунку 1.2 наведено один з можливих розкладів виконання робіт A, B, C і показані відповідні шукані величини

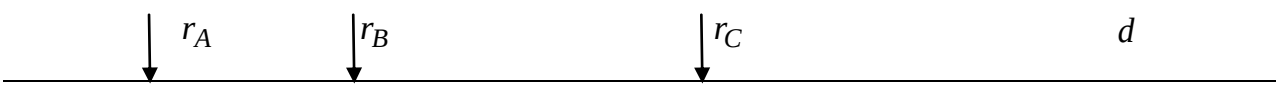


Рисунок 3.1 – Розклад виконання робіт A, B, C

Висновок до розділу

В даному розділі було сформульовано змістовну постановку та математичну постановки задачі, також обгрунтовано метод розв’язання даної задачі та описані методи розв’язання для впорядкування розкладу календарного плану.

У розділі змістовної постановки задачі було описано задачу впорядкування завдань. У розділі математичної постановки задачі описані вхідні дані, критерій оцінювання та шукану цільову функцію.

У розділі обґрунтування методу розв’язання описано обраний метод розв’язання, наведені умови для впорядкування розкладу.

У розділі опису методів розв’язання було представлено та розв’язано задачу впорядкування розкладу. Наведено один з можливих шуканих розкладів.

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Засоби розробки

Під час виконання дипломного проекту були використані такі технології, як:

- мову програмування Javascript;
- бібліотеку React;
- платформу Node.js;
- бібліотеку C3.js та D3.js;
- препроцесор SCSS.

Для зберігання інформації по користовачеві було використано нереляційну базу даних MongoDB.

Увесь процес розробки програмного продукту виконувався за допомогою:

- текстовий редактор Sublime Text 3;
- командний рядок Linux;
- графічний інтерфейс для MongoDB 3T Studio.

Javascript – популярна мова программування, яка широко використовується у розробці інтерактивних веб-сайтів з боку клієнтської частини розробки.

На Javascript написано безліч модулів, бібліотек та фреймворків, які створюють велику кількість варіацій для розробки веб-застосунків.

На сьогоднішній день найчастіше використовуються такі бібліотеки та фреймворки, як:

- JQuery – бібліотека, для полегшення програмування на Javascript;
- Bootstrap – фреймворк для створення веб-застосунків, що мають не важкий у реалізації дизайн та полегшення адаптивності під будь-які пристрої;

- React – бібліотека для створення Sigle Page Applications (далі SPA);
- Angular – більш функціональний, ніж React.js, фреймворк, для розробки SPA.

Для створення та розробки системи підтримки розвитку та самоорганізації особистості було використано React.js через те, що його можливостей вистачило для виконання усіх функціональних умов дипломного проектух.

React — це декларативна, ефективна і гнучка JavaScript-бібліотека, призначена для створення інтерфейсів користувача. Вона дозволяє компонувати складні інтерфейси з невеликих окремих частин коду — “компонентів” [12].

Основні модулі, які були використані у проекті є:

- React-router-dom – це модуль для навігації між компонентами та роботи з Document Object Model (далі DOM). Він надає змогу переходити між компонентами без перезагрузки сторінки та керувати поведінкою елементів HTML розмітки напряму;
- Babel – Javascript-компілятор для перетворення синтаксичного цукру React, ES6, ES7 до ES5;
- Express – модуль для полегшення написання серверної частини розробляємої системи на Node.js;
- Mongoose – модуль для полегшення зв’язування серверної частини проекту та бази даних MongoDB;
- Node-sass – модуль для полегшення написання стилів CSS за допомогою препроцесора SCSS;
- Webpack – модуль за збирання проекту та його повноцінне функціонування, яки відповідає за роботу усіх інших модулів.

Node.js – це платформа для серверної розробки веб-застосунків, яка базується на “двигуні” Google V8. Після того, як розробник написав код на Node.js, V8 компілює його у машинний код для подальшого використання [13].

Ця платформа вміє використовувати сторонні бібліотеки, працювати з файлами у системі, а також виконувати роль веб-серверу для веб-застосунків.

Перевагою Node.js над усіма іншими платформами для серверної розробки є те, що Node.js працює асинхронно. Тобто, якщо багато користувачів матимуть намір одночасно відвідати який-небудь сайт, то дана платформа розтавляє приорітети та розподілює ресурси раціональніше, уа відміну від Java, яка під кожне підключення створює окремий поток.

Для збереження даних про користувача було використано MongoDB. Це нереляційна документно-орієнтована база даних, що була створена з метою бути більш гнучкою, легко маштабуючою та швидкооброблюваною, ніж її реляційні аналоги, а також для забезпечення високої доступності, підтримує динамічні схема та дозволяє легко перерозприділяти дані між декількома серверами [14].

Для полегшення доступу до MongoDB у застосунку використовували модулі для Node.js, такі як: Express та Mongoose.

4.2 Вимоги до технічного забезпечення

Для раціональної та повноцінної роботи системи необхідним технічним засобом є комп’ютер зі встановленим браузером та можливістю виходу в Інтернет.

4.3 Архітектура програмного забезпечення

Програмний продукт містить у собі 2 застосунки:

- Web API застосунок «backend-api» для збереження даних у БД, обробки даних для передачі їх до застосунку інтерфейсу користувача у оптимізованому вигляді задля полегшення роботи веб-браузера;
- React-застосунок інтерфейсу користувача «client-structure-interface».

Перевагами такої архітектури «клієнт-сервер» пов’язані з тим, що клієнтська частина відповідає лише за відображення, та не несе у собі ніякої складної логіки в порівнянні з серверною частиною, яка працює з конфіденційно-важливими даними. Спілкування між React застоунком на клієнті та API застосунком на сервері відбувається лише за допомогою AJAX – запитів, або XML HTTP запитів (скорочено XHR). Така архітектура дозволяє з легкістю забезпечити безпеку даних користувача, та надає змогу швидше оброблювати та предоставляти дані для користувача.

Web API застосунок «backend-api» - створений на основі REST архітектури, що дозволяє використовувати такі види запитів до серверної частини від клієнта, такі як:

- GET - повертання URI та деякі інші деталі ресурсів колекції;
- POST - створення нового ресурсу в колекції;
- PUT - заміна однієї колекції деякою інакшою;
- DELETE – видалення усієї колекції [15].

Даний застосунок розроблений за допомогою трьохрівневої архітектури. Цей тип архітектури широко використовується для створення программного забезпечення клієнт-серверної системи.

Рівні серверної частини:

- web API – приймає вхідні данні за допомогою REST-запитів, та повертає дані до клієнтської частини програмного продукту у вигляді JSON або HTML;

– рівень бізнес-логіки, яка надає змогу керувати усіма створеними можливостями програми.

4.3.1 Діаграма взаємодії

Розглянемо детальніше взаємодію компонентів у «backend-api», яка наведена у графічному матеріалі.

Дана структурна схема демонструє зв’язок між пакетами NODE js, які були використані для створення підключення до БД, отримання даних з клієнтської частини застосунку та передачі на клієнтську частину даних.

Для зберігання даних користувача у БД було використано mongoose бібліотеку, задля створення запитів напряму від серверу до БД. Mongoose надає змогу легко та швидко підключатися, записувати та діставати потрібні дані з MongoDB.

У схемі бази даних зображено зв’язок та залежність між схемами(моделями), що були використані для розуміння базою даних, яку структуру їй необхідно створити. Ця схема надає опис інтерфейсів для структури бази даних.

Розглянемо детальніше моделі, що були проілюстровані у схемі класів.

– User – схема, для опису користувача системи. Містить у собі ідентифікаційні дані користувача для авторизації його в системі;
– Categories – схема для опису кожної сфери колеса життя, що містить у собі дані, для поставлених цілей, оцінок користувача по обраній сфері а також інші дані, необхідні для користування програмним продуктом.

У серверному застосунку створені сервіси та контролери, що приймають та відправляють REST запити за допомогою HTTP протоколу.

4.3.2 Діаграма послідовності

Для більш якісного розуміння внутрішніх процесів передачі даних між застосунками та класами розглянемо, як реагує система на такі дії від користувача, як реєстрація та заповнення оцінок сфер колеса життя.

Процеси реєстрації користувача наведені у графічному матеріалі.

Процес заповнення оцінок сфер колеса життя наведені у графічному матеріалі.

4.4 Специфікація функцій

Розглянемо специфікацію публічних функцій (публічний API) застосувань.

Таблиця 4.1 – Функції класів програмного забезпечення

Назва	Опис
Застосунок «backend-api» «server.js»:	
app.listen(port, '0.0.0.0', (err))	Метод, що створює та підіймає серверну частину програмного продукту.
app.use(webpackDevMiddleware(compiler,{…})	Функція запуску та відслідковування змін клієнтської частини за допомогою Webpack без перезавантаження серверної частини програмного продукту.
Застосунок «backend-api» клас «user.js»:	
app.get('/api/user/:username:password, (req, res) => {})	REST GET запит для підтвердження авторизації користувача в системі. Повертає інформацію про користувача, або повідомлення про те, що такого користувача не знайдено
app.post('/api/user/registration, (req, res) => {})	REST POST запит для підтвердження авторизації користувача в системі. Повертає повідомлення про успішну або невдалу спробу запису інформації про користувача у БД

Продовження таблиці 4.1

Застосунок «backend-api» клас «lifediagram.js»:	
app.post('/api/lifewheel/create, (req, res) => {…})	REST POST запит для створення колекції оцінок по по всім сферам колеса життя. Приймає колекцію оцінок та сфер життя та вносить їх до нового документу у БД. Повертає повідомлення про успішну або невдалу спробу запису інформації у БД
app.post('/api/lifewheel/show, (req, res) => {…})	REST POST запит для діставання колекції оцінок по всім сферам колеса життя. Приймає об'єкт з даними користувача. Повертає колекцію оцінок користувача по всім сферам колеса життя або повідомлення про невдалу спробу діставання інформацію з бд.
Застосунок «backend-api» клас «statistic.js»:	
app.post('/api/statistics/show, (req, res) => {…})	REST POST запит для діставання оцінок користувача з БД за увесь час спринту по певній сфері. Приймає об'єкт з інформацією про користувача, номером сфери. Повертає колекцію оцінок певної сфери колеса життя або повідомлення про невдалу спробу запису інформації у БД
Застосунок «backend-api» клас «calendar.js»:	
app.put('/taskSave', function (req, res) {});	REST PUT запит для створення завдання по певній сфері життя. Приймає об'єкт з інформацією про користувача, номером сфери, даними по завданню. Записує у бд інформацію про завдання згідно отриманими даними Повертає повідомлення про успішну або невдалу спробу запису інформації про завдання у БД

Продовження таблиці 4.1

app.post('/alltasks, function (req, res) {});	REST POST запит для діставання з БД інформації по завданням, цілям та підцілям. Приймає об’єкт з інформацією про користувача Повертає колекцію з даними по завданням, цілям, підцілям або повідомлення про невдалу спробу діставання інформації з БД

Висновок до розділу

У розділі програмного та технічного забезпечення були визначенні технології, які були використані для створення та розробки програмного продукту. Були описані переваги цих технологій та обгрунтовано їх вибір.

У розділі архтектура програмного забезпечення було наведено опис структурних схем класів та компонентів застосунків, що були створенні для вирішення усіх функціональних та нефункціональних вимог до програмного продукту. Описано основні бізнес процеси – реєстрація користувача у системі та заповнення оцінок сфер колеса життя. Ці бізнес процеси були представлення у вигляді взаємодії між класами та проілюстровані на схемах послідовності. Так як системна архітектура відповідає архітектурі «клієнт-сервер», то обмін даними між застосунками здійснюється за допомогою REST запитів, що використовують HTTP протокол передачі даних і представлений на схемах.

Опис архітектури представлений для зостусунку «backend-api». На схемі класів бізнес сутностей наведена структура схем, що відповідає структурі БД.

Даний розділ містить специфікацію функцій, що використані у серверній частині програмного продукту. Саме ці функції, доступні користувачам програмного продукту та описують взаємодію через REST запити між клієнтською та серверною частиною прогамного продукту.

5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

5.1 Керівництво користувача

Для початку роботи з системою користувач мусить відкрити вікно браузеру, та перейти за шляхом http://localhost:3000/. Після того, як з серверу підгрузяться дані, необхідні для відображення сторінки, користувач матиме змогу взаємодіяти із головно сторінкою, яка представлена на рисунку 5.1

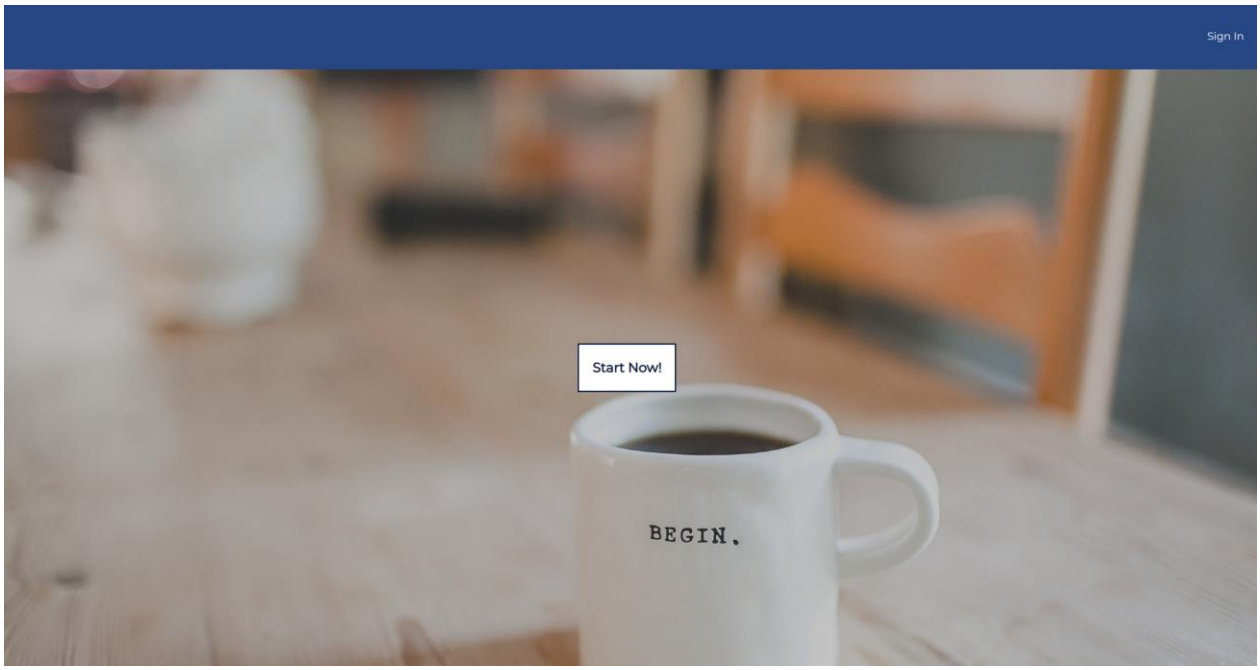


Рисунок 5.1 – Головна сторінка

Як ми можемо побачити, то функціоналу для користувача, який не авторизований у системі обмежено:

— авторизацією – посилання на сторінку авторизації (Напис Sign in на рисунку);

— реєстрацією – посилання на сторінку реєстрації (Напис Start Now!).

Перейдемо на сторінку реєстрації та введемо усі необхідні дані (рисунок 5.2).

Рисунок 5.2 – Сторінка реєстрації

На сторінці реєстрації у нас присутні такі поля, як:

— почтова адреса;

— пароль;

— підтвердження паролю.

Якщо, якесь поле заповнене неправильно, або з можливістю угрози системі, то кнопка Submit буде неактивною до тих пір, поки усі дані не будут введені коректно.

Після того, як було введено коректні дані, користувач може зареєструватися та перейти на сторінку створення діаграми колеса життя (рисунок 5.3).

Рисунок 5.3 – Сторінка створення колеса життя

На даному рисунку можна побачити, що активною сторінкою є WheelLife, усі інші сторінки для користувача зараз недоступні.

Користувач має можливість оцінити теперішній стан розвитку кожної сфери колеса життя від 1 до 10, вибираючи у полі зі списком Mark певну оцінку.

Після того, як було обрана оцінка, кнопка Next section разблокується і користувач матиме змогу перейти на оцінювання іншої сфери колеса життя (рисунок 5.4).

Рисунок 5.4 – Заповнення останньої сфери колеса життя

Після оцінювання останньої сфери, користувачеві надається можливість створити свою діаграму колеса життя.

Для цього йому потрібно нажати на кнопку Create your wheel, після чого система автоматично згенерує та відобразить діаграму колеса життя (рисунок 5.5).

Рисунок 5.5 – Сформоване колесо життя

На даному этапі ми бачимо діаграму розвитку сфер на теперішній стан. Звідси ми можемо зрозуміти, які сфери життя розвинуті найгірше.

З лівого боку можна побачити навігацію по системі, що скадається з таких посилань, як:

— LifeWheel – посилання, на сторінку відображення колеса життя;
— Statistic – посилання, на сторінку формування статистики по певній сфері;
— Aims – посилання, на сторінку вибору цілей для кожної сфери;
— Calendar - посилання, на сторінку календаря, що заповнений завданнями для кожної цілі на певний проміжок часу.

Розглянемо перехід на сторінку формування статистики по певній сфері (рисунок 5.6).

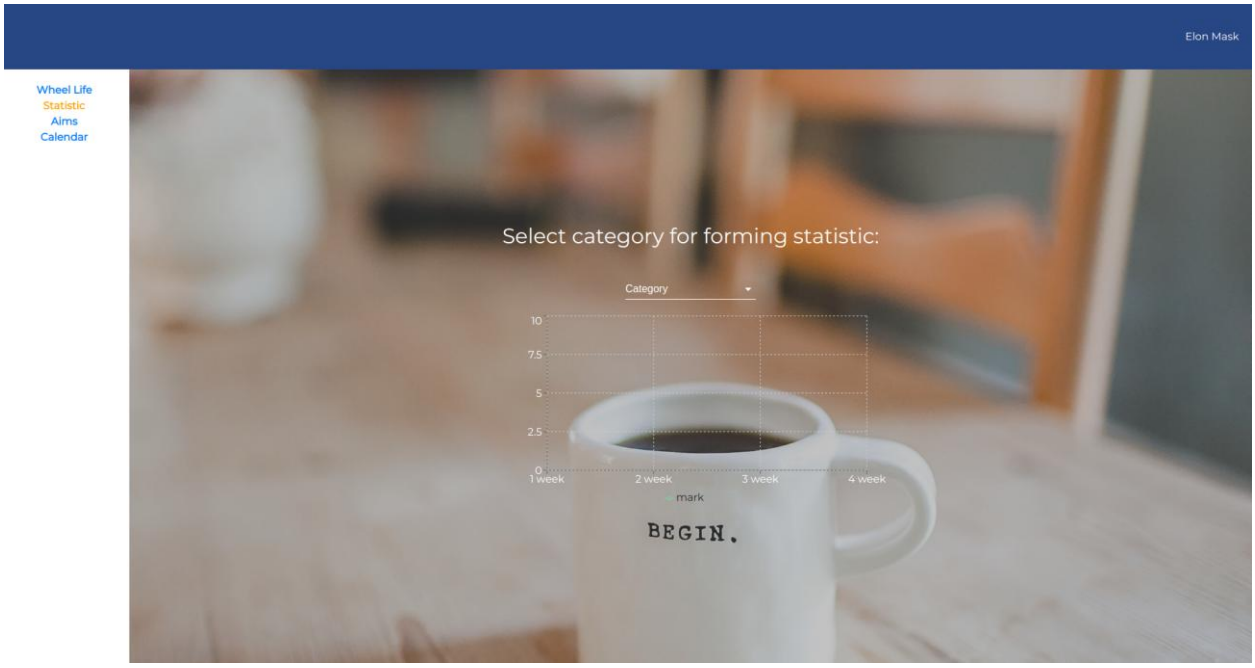


Рисунок 5.6 – Сторінка формування статистики по певній сфері

У полі зі списком Category, користувач обирає, по якій сфері колеса життя буде сформована статистика. Після вибору значення у цьому полі зі списком, система автоматично згенерує та відобразить діаграму статистики по заданій сфері (рисунок 5.7).

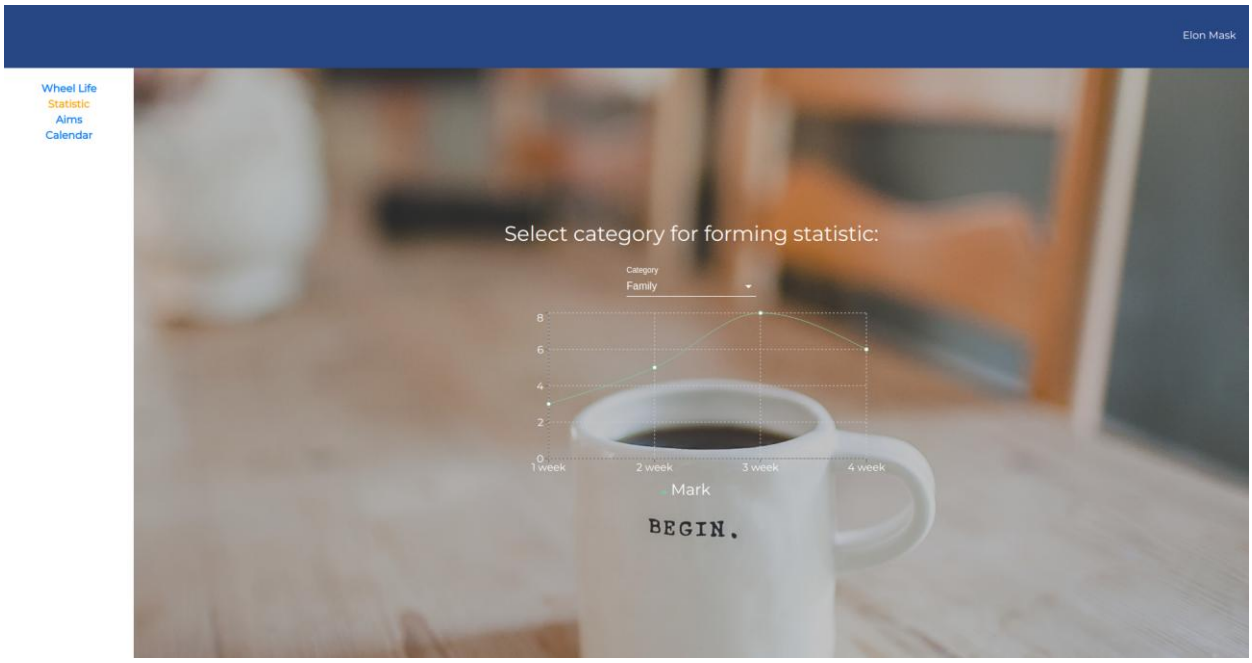


Рисунок 5.7 – Сформована статистика по певній сфері.

Як можна побачити з рисунку 5.8, дана діаграма дозволяє відслідковувати зміну оцінки сфери колеса життя, впродовж певного проміжку часу, а саме 4 тижні.

Перейдемо до сторінки формування цілей (рисунок 5.8).

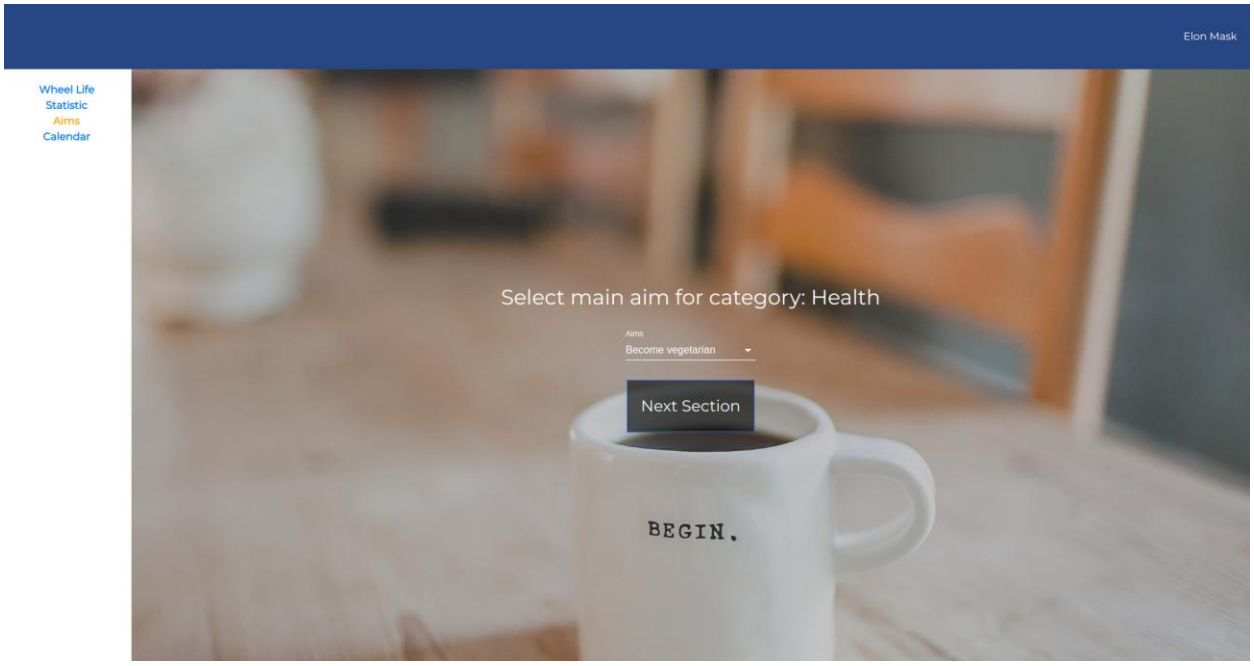


Рисунок 5.8 – Сторінка вибору цілей, по певній сфері

У полі зі списком обирається ціль, яку користувач прагне досягнути до кінця спринта.

Після вибору певної цілі, користувач переходить до вибору наступної цілі за наступною сферою, за допомогою кнопки Next Section (рисунок 5.9).

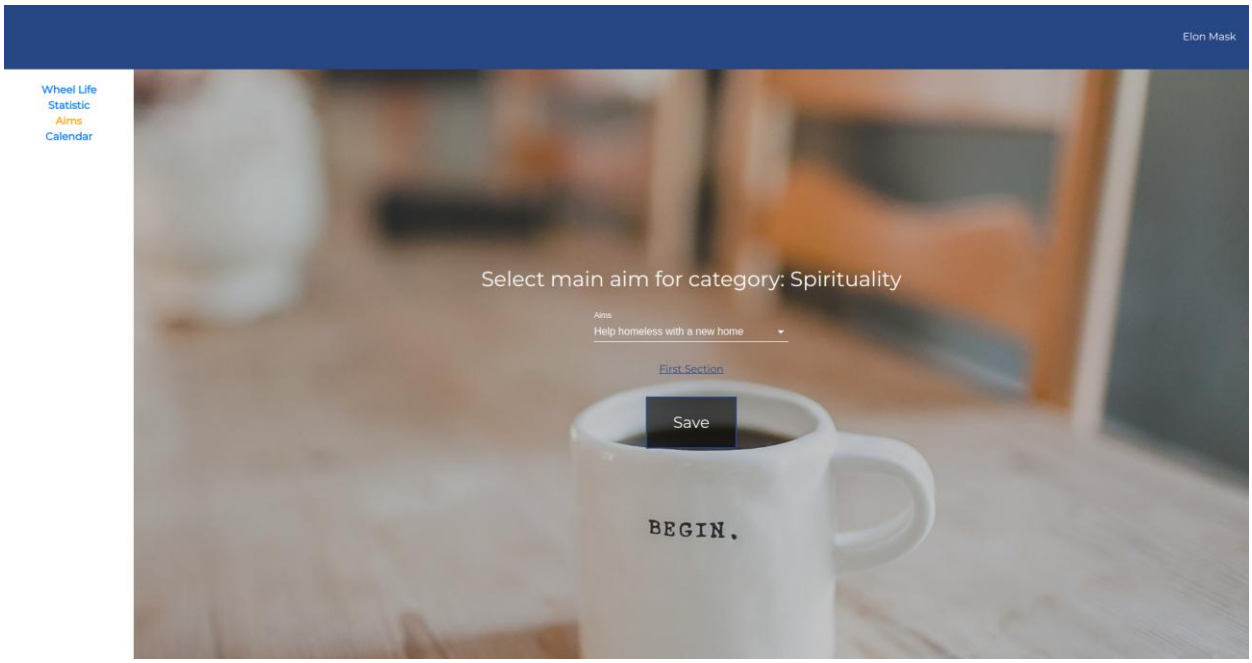


Рисунок 5.9 – Вибір останньої цілі для залишеної сфери.

Користувачеві надається змога повернутися до початку заповнення усіх цілей, якщо він натисне на посилання First Section.

Якщо ж користувач усе заповнив, згідно його переваг та намірів, то система зберігає дані про цілі у БД.

Перейдемо на сторінку Календарного плану, за допомогою посилання Calendar (рисунок 5.10).

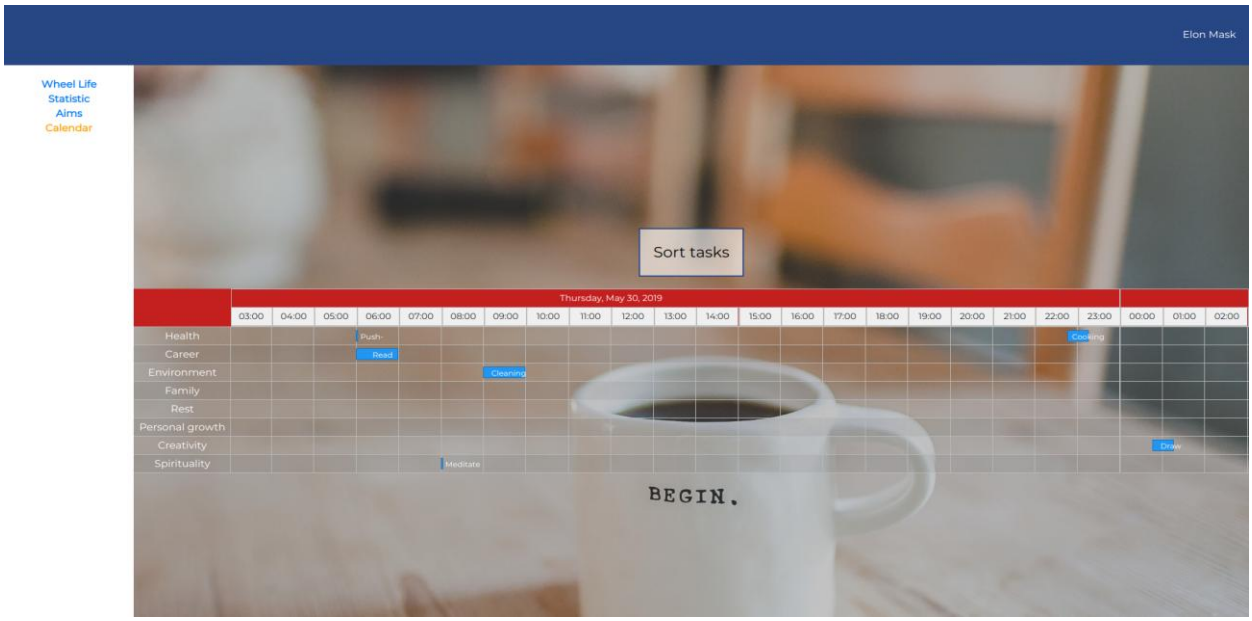


Рисунок 5.10 – Сторінка календарного плану

Як можна побачити на рисунку 5.10 ми маємо таблицю завданнь, де кожен рядок – це сфера колеса життя, а кожний стовпець – це певний час і дата.

Користувач має змогу додавати нове завдання, просто натиснувши на відповідну комірку у календарі. Також є можливість змінювати вручну тривалість виконання справи, встановлені сроки та опис завдання.

Усі зміни автоматично зберігаються у системі.

Натиснемо на кнопку Sort Tasks та отримаємо новий перетворений та оптимізований розклад, за алгоритму впорядкування розкладу за критерієм своєчасного виконання робіт (рисунок 5.11).

Рисунок 5.11 – Відсортований календарний план

Як можна побачити з рисунку 5.11 – відображається новий розклад, завдяки якому користувач має змогу зробити усі завдання за менший термін.

Розглянемо ситуацію, коли користувач вже зареєстрований у системі. Перейдемо з головної сторінки на сторінку авторизації користувача (рисунок 5.12).

Рисунок 5.12 – Сторінка авторизації

Необхідні поля для заповнення це:

— поштова адреса користувача;

— пароль користувача;

Після заповнення цих данних, користувач має змогу перейти на сторінку вже побудованої діаграми, а то використовувати функції системи так, як і тільки-но зареєстрований користувач, окрім заповнення цілей. Щоб мати змогу заповни

5.2 Випробування програмного продукту

Розглянемо методики випробувань програмного продукту, порядок випробувань та опис тестів для перевірки відповідності програмного продукту функціональним вимогам, що описані у технічному завданні

5.2.1 Мета випробувань

Метою випробувань являється перевірка відповідності функцій комплексу задач Інформаційна система підтримки розвитку та самоорганізації особистості вимогам технічного завдання.

5.2.2 Загальні положення

Випробування проводяться на основі наступних документів:

– ГОСТ 34.603–92. Інформаційна технологія. Види випробувань автоматизованих систем;

– ГОСТ РД 50-34.698-90. Автоматизовані системи вимог до змісту документів.

5.2.3 Результати випробувань

Кожен застосунок програмного продукту був перевірений на відповідність функціональним вимогам. Зміст та результати випробувань розглянуті нижче у таблицях 5.1 – 5.10.

Розглянемо зміст та результати тестування. Тести включають перевірку основної функціональності – навігація сайтом, авторизація та реєстрація, аналіз зображень та збереження результатів. Тестові сценарії та результат їх проходження наведений у таблицях 5.1 – 5.7.

Таблиця 5.1 – Тестування форми авторизації на відкриття сторінки діаграми колеса життя

Тест:	Форма авторизації при успішній авторизації відкриває сторінку діаграми колеса життя
Початковий стан системи:	Відкрита сторінка з формою авторизації
Дія:	Вводимо правильний логін та пароль і натискаємо кнопку «вхід»
Очікуваний результат:	Відкривається сторінка діаграми колеса життя; В меню відображається ім’я поточного користувача
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.2 – Тестування меню користувача – гостя

Тест:	В меню користувача відображені посилання на доступні незареєстрованому користувачу системи
Початковий стан системи:	Відкрита домашня сторінка; користувач не ввійшов в систему
Дія:	-
Очікуваний результат:	Користувач бачить кнопку «почати роботу», посилання на сторінку «авторизація»
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.3 – Тестування меню авторизованого користувача

Тест:	В меню користувача відображені посилання на доступні авторизованому користувачу
Початковий стан системи:	Відкрита сторінка діаграми колеса життя; користувач ввійшов в систему
Дія:	-
Очікуваний результат:	Користувач бачить назву застосунку, посилання на сторінки «Статистика», «Цілі», «Колесо Життя», «Календар» та своє ім’я
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.4 – Тестування меню користувача, посилання на сторінку статистики

Тест:	Посилання на сторінку статистики в меню користувача відкриває сторінку статистики
Початковий стан системи:	Відкрита сторінка діаграми колеса життя; користувач ввійшов в систему
Дія:	Натискаємо на посилання «Statistic»
Очікуваний результат:	Перехід на сторінку статистики
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.5 – Тестування меню користувача, посилання на сторінку заповнення цілей

Тест:	Посилання на сторінку заповнення цілей в меню користувача відкриває сторінку заповнення цілей
Початковий стан системи:	Відкрита сторінка діаграми колеса життя; користувач ввійшов в систему
Дія:	Натискаємо на посилання «Aims»
Очікуваний результат:	Перехід на сторінку заповнення цілей
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.6 – Тестування меню користувача, посилання на календарного плану

Тест:	Посилання на сторінку календарного плану в меню користувача відкриває сторінку календарного плану
Початковий стан системи:	Відкрита сторінка діаграми колеса життя; користувач ввійшов в систему
Дія:	Натискаємо на посилання «Calendar»
Очікуваний результат:	Перехід на сторінку календарного плану
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.7 – Тестування перестановки розкладу для авторизованого користувача

Тест:	На сторінці календарного плану авторизований користувач має змогу впорядкувати свій розклад
Початковий стан системи:	Відкрита сторінка календарного плану; користувач ввійшов в систему
Дія:	Натискаємо на посилання «Sort Schedule»
Очікуваний результат:	На сторінці відображається новий календарний план вже з відсортованим списком завдань
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.8 – Тестування створення нового завдання для авторизованого користувача

Тест:	На сторінці календарного плану авторизований користувач має змогу створити нове завдання до вже розробленого календарного плану
Початковий стан системи:	Відкрита сторінка календарного плану; користувач ввійшов в систему
Дія:	Натискаємо на пусту комірку у календарному плані
Очікуваний результат:	На сторінці відображається модальне вікно для створення нового завдання
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.9 – Тестування додавання нового завдання для авторизованого користувача

Тест:	На сторінці календарного плану авторизований користувач має змогу додати завдання до вже розробленого календарного плану
Початковий стан системи:	Відкрита сторінка календарного плану; заповнене модальне вікно для нового завдання; користувач ввійшов в систему
Дія:	Натискаємо на кнопку «Add new task»
Очікуваний результат:	На сторінці відображається новий календарний план вже з доданим завданням
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.10 – Тестування відміни додавання нового завдання для авторизованого користувача

Тест:	На сторінці календарного плану авторизований користувач має змогу відмінити додавання завдання до вже розробленого календарного плану
Початковий стан системи:	Відкрита сторінка календарного плану; відображено модальне вікно для нового завдання; користувач ввійшов в систему
Дія:	Натискаємо на кнопку «X» на модальному вікні
Очікуваний результат:	На сторінці відображається не зміненний календарний план
Фактичний результат співпадає з очікуваним:	Так

Висновок до розділу

У технологічному розділі було розглянуто детально опис програмного продукту та представлені його екранні форми.

У розділі керівництво користувача розглянуто детально користувацький інтерфейс. Представлений опис меню користувача – призначення кожного посилання в меню, процес реєстрації та авторизації користувача. Було розглянуто сторінки створення діаграми колеса життя, вибору цілей, відображення статистики та відображення календарного плану. Для сторінки календарного плану було розглянуто процес впорядкування розкладу

У розділі випробування програмного продукту наведено мету та загальні положення випробувань. У розділі наведені тестові сценарії, що перевіряють відповідність програмного продукту функціональним вимогам, представленим у технічному завданні.

ЗАГАЛЬНІ ВИСНОВКИ

У пояснювальный записці розглянуто детальний опис дипломного проекту, що присвячений розробці інформаційної системи підтримки розвитку та самоорганізації особистості

У розділі опису загальних положень було визначено мету та призначення програмного продукту.

У розділі опису предметного середовища описано предметну область дипломного проекту. Для реалізації алгоритму впорядкування розкладу потрібно розуміти, що таке поняття теорії розкладу, які методи впорядкування і при яких умовах використовуться. У цьому розділі описано важливість структуризації даних по сферам життя.

Розглянуто процес діяльності користувача та описано функціональну модель за допомогою схеми варіантів використання – дії, що можуть виконувати певну роль у системі. Саме ця схема відображає функціональні вимоги до системи.

Порівняння системи з іншими застосунками, пояснення чому розроблений програмний продукт є найкращим рішенням серед усіх існуючих.

Розділ інформаційного забезпечення призначений для опису даних, якими оперує система та структур зберігання даних. Описано вхідні дані системи – колекції Users та Categories. Також у розділі вхідних даних описано атрибути колекцій Users та Categories,

У розділі вихідних даних описані дані, що отримує користувач у результаті аналізу, що використовуються для відображення певних діаграм та календаря зі списком завдань. Опис структури бази даних, колекцій та документів представлений у розділі опис структури бази даних.

Розділ математичного забезпечення містить змістовну та математичну постановку задачі. Також він описує обраний алгоритм для розвязання задачі та ілюструє приклад його застосування.

Опис та обгрунтування вибору засобів розробки наведено у розділі програмне та технічне забезпечення. Програмний продукт є інформаційною системою, доступ до якої відбувається через браузер і вимагає певного технічного забезпечення, яке описано у відповідному розділі. У даному розділі описано архітектуру програмного забезпечення: представлено детальний опис застосунків, що були створені для розвязання задачі, описано схему компонентів, схему залежностей застосунків. Основні процеси у системі та демонстрація внутрішньої реалізації та взаємодії окремих класів представлена за допомогою схем послідовності реєстрації користувача та схеми послідовності вибору оцінки згідно вибраної сфери архітектурного стилю та збереження результату. Для публічних інтерфейсів програми наведена специфікація функцій. У технологічному розділі наведено керівництво користувача та екранні форми застосунку. Основні тестові сценарії та результат проходження тестів описаний у розділі випробування програмного продукту.

Програмний продукт відповідає функціональним вимогам, описаним у технічному завданні до дипломного проекту.

ПЕРЕЛІК ПОСИЛАНЬ

1. Мрочковский Н.О., Толкачев О.И. Экстремальный тайм-менеджмент. – Альпина Паблишер, 2015. – 214 с.
2. Google Calendar . [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Google_Calendar
3. Google Calendar . [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Google_Calendar
4. Google . [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Google_Calendar
5. TickTick – совершенній планировщик задач и хранитель заметок. [Електронний ресурс]. – Режим доступу: https://lifehacker.ru/ticktick/
6. TickTick: список дел и управление задачами. [Електронний ресурс]. – Режим доступу: https://play.google.com/store/apps/details?id=com.ticktick.task&hl=ru
7. AnyDo. [Електронний ресурс]. – Режим доступу: https://www.any.do/ru/
8. AnyDo: список задач, календарь, напоминания. [Електронний ресурс]. – Режим доступу: https://play.google.com/store/apps/details?id=com.anydo&hl=ru
9. Time Buddy – Clock&Converter. [Електронний ресурс]. – Режим доступу: https://play.google.com/store/apps/details?id=com.helloka.worldtimebuddy&hl=en_US
10. Hours Time Tracking. [Електронний ресурс]. – Режим доступу: https://itunes.apple.com/us/app/hours-time-tracking/id895933956?mt=8
11. Жданова О.Г. Конспект лекцій з теорії розкладів – 2019. – 76 с.
12. React.js [Електронний ресурс]. – Режим доступу: https://uk.reactjs.org/tutorial/tutorial.html#what-is-react

13. Node.js [Електронний ресурс]. – Режим доступу: https://nodejs.org/uk/about/
14. MongoDB [Електронний ресурс]. – Режим доступу: https://proselyte.net/tutorials/mongodb/advantages/
15. REST [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/REST .

Додаток А

Тексти програмного коду

Інформаційна система підтримки розвитку та самоорганізації

(Найменування програми (документа))

DVD-R

(Вид носія даних)

68 арк, 880 Кб

(Обсяг програми (документа) , арк.,) Кб)

Київ – 2019 року

App.js

```
import React, {Component} from 'react';
import './App.scss';
import Home from './Home';
import Registration from './Registration';
import SignIn from './signin';
import Cabinet from './Cabinet';
import Statistic from './Statistic';
import Aims from './Aims';
import Calendar from './Calendar';
import {
  BrowserRouter as Router,
  Route,
  Link,
  Redirect
} from "react-router-dom";
export default class App extends Component {
  constructor(props){
    super(props);
    this.state = {}
  }
  render() {
    return (
        <Router>
          <div className="App">
              <header className="header">
                <nav className="menu__navigation">
                  <div className="menu__item">
                    <Link to="/signIn" className="menu__item-link">Elon
Mask</Link>
                  </div>
                </nav>
              </header>
              <Route exact path="/" component={Home} />
              <Route path="/signUp" component={Registration} />
              <Route path="/signIn" component={SignIn} />
              <Route path="/mycabinet" component={Cabinet} />
              <Route path="/statistic" component={Statistic} />
              <Route path="/aims" component={Aims} />
              <Route path="/calendar" component={Calendar} />
          </div>
        </Router>
    )
  }
}
```

Registration.js

```
import React, {Component} from "react";
import {
    Redirect
} from "react-router-dom";
import {
    Form, Button
} from "react-bootstrap";
export default class Registration extends Component {
    constructor(props){
        super(props);
        this.state = {
            redirect: false
        }
    }
    setRedirect = () => {
        this.setState({
          redirect: true
        })
      }
    renderRedirect = () => {
    if (this.state.redirect) {
        return <Redirect to='/mycabinet' />
    }
    }
    render() {
        return (
            <main>
                <h1 className="headerTitle">Registration</h1>
                <Form className="LoginForm">
                    <Form.Group controlId="formBasicEmail">
                        <Form.Label>Email address</Form.Label>
                        <Form.Control type="email" placeholder="Enter email" />
                        <Form.Text className="text-muted">
                        We'll never share your email with anyone else.
                        </Form.Text>
                    </Form.Group>
                    <Form.Group controlId="formBasicEmail">
                        <Form.Label>Password</Form.Label>
                        <Form.Control type="password" placeholder="Password" />
                    </Form.Group>
                    <Form.Group controlId="formBasicEmail">
                        <Form.Label>Repeat Password</Form.Label>
                        <Form.Control type="password" placeholder="Password" />
                    </Form.Group>
                    <Form.Group controlId="formBasicChecbox">
                        <Form.Check type="checkbox" label="Check me out" />
                    </Form.Group>
                    {this.renderRedirect()}
                    <Button variant="primary" type="submit"
onClick={this.setRedirect}>
                        Submit
                    </Button>
                </Form>
            </main>
        );
    }
}
```

Calendar.js

```
import React, { Component } from 'react';
import CalendarContainer from './calendar/calendar'
import Sidebar from './sidebar';
import moment from 'moment'
export default class Calendar extends Component {
       constructor(props){
             super(props);
    this.state = {
        groups: [
            { id: 1, title: 'Health' },
            { id: 2, title: 'Career' },
            { id: 3, title: 'Environment' },
            { id: 4, title: 'Family' },
            { id: 5, title: 'Rest' },
            { id: 6, title: 'Personal growth' },
            { id: 7, title: 'Creativity' },
            { id: 8, title: 'Spirituality' },
        ],
        pageName: "calendar",
        items: [
            {
              id: 8,
              group: 8,
              title: 'Meditate',
              start_time: moment().startOf('day').add(8,'hour'),
              end_time: moment().startOf('day').add(20,'minute'),
              itemProps: {
                  // these optional attributes are passed to the root <div /> of
each item as <div {...itemProps} />
                  'data-custom-attribute': 'Random content',
                  'aria-hidden': true,
                  onDoubleClick: () => { console.log('You clicked double!') }
              }
            },
            {
              id: 2,
              group: 2,
              title: "Read economic's book",
              start_time:  moment().startOf('day').add(6,'hour'),
              end_time: moment().startOf('day').add(7,'hour'),
              itemProps: {
                  // these optional attributes are passed to the root <div /> of
each item as <div {...itemProps} />
                  'data-custom-attribute': 'Random content',
                  'aria-hidden': true,
                  onDoubleClick: () => { console.log('You clicked double!') }
              }
            },
            {
              id: 1,
              group: 1,
              title: 'Push-up 20 times',
              start_time: moment().startOf('day').add(6,'hour'),
              end_time: moment().startOf('day').add(6,'hour'),
              itemProps: {
                  // these optional attributes are passed to the root <div /> of
each item as <div {...itemProps} />
                  'data-custom-attribute': 'Random content',
                  'aria-hidden': true,
                  onDoubleClick: () => { console.log('You clicked double!') }
              }
            },
            {
                id: 3,
                group: 3,
```

```
                        title: 'Cleaning workspace',
                        start_time: moment().startOf('day').add(9,'hour'),
                        end_time: moment().startOf('day').add(10,'hour'),
                        itemProps: {
                            // these optional attributes are passed to the root <div />
of each item as <div {...itemProps} />
                            'data-custom-attribute': 'Random content',
                            'aria-hidden': true,
                            onDoubleClick: () => { console.log('You clicked double!') }
                        }
                      },{
                        id: 7,
                        group: 7,
                        title: 'Draw picture',
                        start_time: moment().add(18, 'hour'),
                        end_time: moment().add(18.5, 'hour'),
                        itemProps: {
                            // these optional attributes are passed to the root <div />
of each item as <div {...itemProps} />
                            'data-custom-attribute': 'Random content',
                            'aria-hidden': true,
                            onDoubleClick: () => { console.log('You clicked double!') }
                        }
                      },{
                        id: 4,
                        group: 1,
                        title: 'Cooking salad',
                        start_time: moment().add(16, 'hour'),
                        end_time: moment().add(16.5, 'hour'),
                        itemProps: {
                            // these optional attributes are passed to the root <div />
of each item as <div {...itemProps} />
                            'data-custom-attribute': 'Random content',
                            'aria-hidden': true,
                            onDoubleClick: () => { console.log('You clicked double!') }
                        }
                      },
                    ]
                  };
                }
                render(){
            const props = this.props;
              return(
                <main className="cabinet__wrapper">
                    <Sidebar currentPage={this.state.pageName}/>
                    <div className="diagram_life">
                    <button onClick={this.optimiseSchedule}>Sort tasks</button>
                    <CalendarContainer items={this.state.items}
groups={this.state.groups}></CalendarContainer>
                    </div>
                </main>
              )
            }
        }
```

Cabinet.js

```
import React, { Component } from 'react';
import Sidebar from './sidebar';
import WheelDiagram from './diagram/wheelDiagram';
export default class Example extends Component {
  constructor(props){
    super(props);
    this.state = {
      dataSave: false,
      buttonClass: 'startCreateDiagram'
    };
    this.startCreateDiagram = this.startCreateDiagram.bind(this);
  }
  startCreateDiagram() {
    this.setState({
      dataSave : !this.state.dataSave,
      buttonClass : 'startCreateDiagram hide'

    });
  }
  render() {
    return(
    <main className="cabinet__wrapper">
      <Sidebar/>
      <WheelDiagram startCreating={this.startCreateDiagram} dataSave =
{this.state.dataSave}/>
    </main>
    )
  }
}
```

Sidebar.js

```
import React from 'react';
import {Link} from "react-router-dom";
export default function sidebar({currentPage}){
    return (<div className="left__column">
    <nav className="cabinet__menu">
      <Link className={currentPage == 'Mycabinet' ? 'active' : ''}
to="/mycabinet">Wheel Life</Link>
      <Link className={currentPage == 'Statistic' ? 'active' : ''}
to="/statistic">Statistic</Link>
      <Link className={currentPage == 'Aims' ? 'active' :
''}to="/aims">Aims</Link>
      <Link className={currentPage == 'Calendar' ? 'active' :
''}to="/calendar">Calendar</Link>
    </nav>
  </div>)
}
```

Aims.js

```
import React, { Component } from 'react';
import FormControl from '@material-ui/core/FormControl';
import Select from '@material-ui/core/Select';
import InputLabel from '@material-ui/core/InputLabel';
import Sidebar from './sidebar';
export default class Statistic extends Component {
       constructor(props){
              super(props);
    this.state = {
        pageName: 'Aims',
    };
       }
       render(){
    const props = this.props;
      return(
    <main className="cabinet__wrapper">
        <Sidebar currentPage={this.state.pageName}/>
        <div className="diagram_life">
            <div className="ChooseCategoryStatistic">
            <h2 className="chooseSection_title">Select main aim for category:
<span>Spirituality</span></h2>
                <FormControl className="select">
                    <InputLabel htmlFor="age-native-simple">Aims</InputLabel>
                    <Select
                    native
                    value={this.state.age}
                    inputProps={{
                        name: 'aims',
                        id: 'aims-native-simple',
                    }}
                    >
                    <option value="" />
                    <option value={1}>Help homeless with a new home</option>
                    </Select>
                </FormControl>
                <button onClick={this.props.startCreating} >First
Section</button>
                <button onClick={this.props.startCreating} >Save</button>
            </div>
        </div>
    </main>
      )
    }
}
```

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”
Кафедра автоматизованих систем обробки інформації та управління

УЗГОДЖЕНО
Керівник проекту
_____ *М.О. Сперкач*
(підпис) (ініціали, прізвище)
“15” квітня 2019 р.

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ *О.А. Павлов*
(підпис) (ініціали, прізвище)
“15” квітня 2019 р.

Інформаційна система підтримки розвитку та самоорганізації особистості

ТЕХНІЧНЕ ЗАВДАННЯ

Шифр *ДП ІС-5224.1181-с.ТЗ*

на 10 сторінках

Київ – 2019 року

ЗМІСТ

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ 3

1.1 Повне найменування системи та її умовне позначення 3

1.2 Найменування організації-замовника та організацій-учасників робіт 3

1.3 Перелік документів, на підставі яких створюється система 3

1.4 Планові терміни початку і закінчення роботи зі створення системи 4

2 ПРИЗНАЧЕННЯ І ЦІЛІ СТВОРЕННЯ СИСТЕМИ 5

2.1 Призначення системи 5

2.2 Мета створення системи 5

3 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 6

3.1 Вимоги до функціональних характеристик 6

3.2 Вимоги до надійності 6

3.3 Умови експлуатації 7

3.4 Вимоги до складу і параметрів технічних засобів 7

4 СТАДІЇ І ЕТАПИ РОЗРОБКИ 9

5 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ СИСТЕМИ 10

5.1 Види випробувань 10

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Повне найменування системи та її умовне позначення

Повне найменування системи: «Інформаційна система підтримки розвитку та самоорганізації особистості».

Коротке найменування системи: «Інформаційна система підтримки розвитку та самоорганізації особистості».

1.2 Найменування організації-замовника та організацій-учасників робіт

Замовником проекту є кафедра Автоматизованих систем обробки інформації та управління факультету інформатики та обчислювальної техніки Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Представник замовника: доцент кафедри АСОІУ Сперкач Майя Олегівна. Адреса замовника: м. Київ, п-кт Перемоги 37.

Розробником системи є студент групи ІС-52 кафедри Автоматизованих систем обробки інформації та управління Національного технічного університету України "Київський політехнічний інститут ім. І. Сікорського" Черков Павло Юрійович.

1.3 Перелік документів, на підставі яких створюється система

При розробці системи і створення проектно-експлуатаційної документації виконавець повинен керуватися вимогами наступних нормативних документів:

- ДСТУ 19.201-78. Технічне завдання. Вимоги до змісту і оформлення;
- ДСТУ 34.601-90. Комплекс стандартів на автоматизовані системи. Автоматизовані системи. Стадії створення;

– ДСТУ 34.201-89. Інформаційні технології. Комплекс стандартів на автоматизовані системи. Види, комплексність і позначення документів при створенні автоматизованих систем.

1.4 Планові терміни початку і закінчення роботи зі створення системи

Плановий строк початку роботи по створенню системи 7 березня 2019 року. Плановий строк кінця роботи по створенню системи для громадської системи фіксації правопорушень та її адміністрування 31 травня 2019 року.

2 ПРИЗНАЧЕННЯ І ЦІЛІ СТВОРЕННЯ СИСТЕМИ

2.1 Призначення системи

Система призначена для підтримки розвитку та самоорганізації особистості. Для кращого відслідковування стану користувача використано методику, що називається «Колесо життя».

2.2 Мета та задачі створення системи

Метою створення системи є підвищення рівня самоорганізації особистості та її розвитку.

Для досягнення поставленої мети мають бути вирішені такі задачі:

- генерація діаграми розвитку сфер життя;
- створення кінцевих цілей для кожної сфери;
- розробка поточного плану спринта;
- формування статистики.

3 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вимоги до функціональних характеристик

Функціональні вимоги до системи:

1. Система надає можливість сформувати діаграму життя згідно з оцінкою кожної сфери.
   1.1. Система надає можливість обрати певну сферу життя.
   1.2. Система надає можливість ввести ціль на спринт для обраної сфери.
   1.3. Система надає можливість ввести завдання для обраної цілі спринт.
   1.4. Система надає можливість користувачеві оцінити поточний стан сфери за шкалою від 1 до 10 (1 – усе погано, 10 – усе неперевершено).
2. Система надає можливість сформувати статистику згідно з даними, заповненими користувачем.
   2.1. Система надає можливість обрати сферу життя (здоров’я, кар’єра, оточення, сім’я та стосунки, відпочинок, особистий ріст, творчість, духовність) для формування статистики.
   2.2. Система надає можливість сформуватиі та відобразити статистику за обраною сферою життя (здоров’я, кар’єра, оточення, сім’я та стосунки, відпочинок, особистий ріст, творчість, духовність).
3. Система надає можливість сформувати розклад за допомогою функції покращення раціональності розкладу.
   3.1. Система надає можливість записати справу на день
      3.1.1. Система надає можливість помітити справу, як справу з жорстко встановленним часом.
      3.1.2. Система надає можливість обрати проміжок часу для виконання справ.

4. Система надає можливість сформувати план спринта
   4.1. Система надає можливіть поставити відмітку того, що справу виконано

3.2 Вимоги до надійності

Програма повинна зберігати працездатність і забезпечувати відновлення своїх функцій при помилках в роботі апаратних засобів (крім носіїв даних і програм).

Продукт повинен поєднувати надійність і функціональність. У разі виникнення аварійних ситуацій необхідно сповіщати користувача та надавати інструкцію для подальших дій. Будь-які аварійні ситуації мають бути задокументовані у звіті, який при необхідності надсилається розробнику для визначення причини збою в роботі та усуненні помилок, які могли привести до нестабільної роботи програмного продукту.

3.3 Умови експлуатації

Для експлуатації розроблюваної системи користувач має мати персональний комп’ютер з встановленим веб-браузером та доступом в інтернет.

Всі користувачі системи повинні дотримуватися правил експлуатації електронної обчислювальної техніки.

3.4 Вимоги до складу і параметрів технічних засобів

Для коректного функціонування системи було виділено наступні вимоги до технічного забезпечення.

До складу технічних засобів повинні входити:

— комп’ютер з такою конфігурацією:

1) процесор з тактовою частотою не нижче 1.4 ГГц;
2) 64 або 32 розрядна операційна система;
3) достатній об’єм оперативної пам’яті (не менше 2 Гб);

4) інші складові можуть мати будь-які параметри, тому що вони не значним чином впливають на роботу програми;

— додатково має бути встановлене таке програмне забезпечення:

1) база даних MongoDB v5.2+;
2) Node.js v.10.1+;
3) Будь-який браузер, окрім IE, Edge;

— комп’ютерна периферія, до складу якої входить:

1) монітор;
2) мишка;
3) клавіатура.

4 СТАДІЇ І ЕТАПИ РОЗРОБКИ

У таблиці 5.1 наведено календарний план робіт та терміни їх виконання.

Таблиця 5.1 – Календарний план виконання робіт

№	Назва етапів виконання дипломного проекту	Срок виконання
1.	Визначення та узгодження теми дипломного проекту	07.03.2019
2.	Опис предметного середовища. Опис процесу діяльності та опис функціональної моделі	10.03.2019
3.	Огляд наявних аналогів	11.03.2019
4.	Узгодження постановки задачі. Визначення призначення, мети та задач розробки	20.03.2019
5.	Визначення засобів розробки	20.04.2019
6.	Визначення та підготовка вхідних даних	25.03.2019
7.	Визначення структури бази даних та створення бази даних	10.04.2019
8.	Описання архітектури програмного забезпечення	11.04.2019
9.	Розробка дизайну веб-застосунку	12.04.2019
10.	Розробка клієнтської частини веб-застосунку	19.04.2019
10.	Розробка серверної частини веб-застосунку	05.05.2019
11.	Підключення бази даних до серверної частини веб-застосунку	15.05.2019
12.	Тестування системи, виявлення та усунення недоліків	20.05.2019

5 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ СИСТЕМИ

5.1 Види випробувань

Види випробувань узгоджуються з замовником до проведення випробувань.

Здача програмного забезпечення відбувається на комп’ютері виконавця завдання з встановленим програмним забезпеченням, що відповідає вимогам.

Всі випробування, які проводилися для перевірки програмного продукту, наведені у пояснювальній записці до дипломного проекту. Методика тестувань наведена детально у пояснювальній записці. Випробуванням підлягає перевірка основних функцій системи. Було проведене функціональне, модульне та інтеграційне тестування.

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”
Кафедра автоматизованих систем обробки інформації та управління

УЗГОДЖЕНО

Керівник проекту

_____ *М.О. Сперкач*
(підпис) (ініціали, прізвище)

“13” травня 2019 р.

ЗАТВЕРДЖУЮ

В.о. завідувача кафедри

_____ *О.А.Павлов*
(підпис) (ініціали, прізвище)

“14” травня 2019 р.

ІНФОРМАЦІЙНА СИСТЕМА ПІДТРИМКИ РОЗВИТКУ ТА САМООРГАНІЗАЦІЇ ОСОБИСТОСТІ

ПРОГРАМА ТА МЕТОДИКА ВИПРОБУВАНЬ

Шифр *ДП ІС-5224.1181-с.ПМВ*

на 9 сторінках

Київ – 2019 року

ЗМІСТ

1 Об'єкт випробування 3
1.1 Найменування програми 3
1.2 Область застосування 3
2 Мета випробувань 3
3 Вимоги до програмного продукту 3
3.1 Вимоги до функціональних характеристик 3
3.1.1 Вимоги до складу виконуваних функцій 4
4 Вимоги до програмної документації 5
5 Склад і порядок випробувань 6
6 Методи випробувань 9

1 ОБ'ЄКТ ВИПРОБУВАННЯ

Об’єктом тестових випробувань є веб-застосунок розпізнавання архітектурних стилів будівель за зображеннями.

1.1 Найменування програми

Повне найменування системи: Інформаційна система підтримки розвитку та самоорганізації особистості

1.2 Область застосування

Документування результатів виконання робіт, візуалізація та структуризація робіт

2 МЕТА ВИПРОБУВАНЬ

Метою випробувань є перевірка відповідності системи підтримки розвитку та самоорганізації особистості вимогам технічного завдання.

3 ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ

3.1 Вимоги до функціональних характеристик

Функціональні характеристики системи мають відповідати функціональним вимогам функціональних характеристик наведених у технічному завданні дипломного проекту.

3.1.1 Вимоги до складу виконуваних функцій

Функціональні вимоги до системи:

1. Система надає можливість сформувати діаграму життя згідно з оцінкою кожної сфери.
   1.1. Система надає можливість обрати певну сферу життя.
   1.2. Система надає можливість ввести ціль на спринт для обраної сфери.
   1.3. Система надає можливість ввести завдання для обраної цілі спринт
   1.4. Система надає можливість користувачеві оцінити поточний стан сфери за шкалою від 1 до 10 (1 – усе погано, 10 – усе неперевершено)
2. Система надає можливість сформувати статистику згідно з даними, заповненими користувачем.
   2.1. Система надає можливість обрати сферу життя (здоров’я, кар’єра, оточення, сім’я та стосунки, відпочинок, особистий ріст, творчість, духовність) для формування статистики
   2.2. Система надає можливість сформуватиі та відобразити статистику за обраною сферою життя (здоров’я, кар’єра, оточення, сім’я та стосунки, відпочинок, особистий ріст, творчість, духовність)
3. Система надає можливість сформувати розклад за допомогою функції покращення раціональності розкладу.
   3.1. Система надає можливість записати справу на день
      3.1.1. Система надає можливість помітити справу, як справу з жорстко встановленним часом
      3.1.2. Система надає можливість обрати проміжок часу для виконання справ
4. Система надає можливість сформувати план спринта

4.1. Система надає можливіть поставити відмітку того, що справу виконано

4 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

Документація програмного забезпечення має включати в себе специфікацію функцій публічного програмного інтерфейсу, технічне завдання, опис обраних технологій, текст програмного коду та випробування програмного продукту.

Специфікація функцій публічного програмного інтерфейсу передбачає опис функцій, їх призначення, можливі варіанти результатів та опис контрактів, через які відбувається взаємодія.

Технічне завдання містить опис системи – найменування системи, найменування замовника, календарний план робіт, постановку задання, функціональні та нефункціональні вимоги до програмного забезпечення. У технічному завданні має бути наведено призначення та мета створення системи.

Опис обраних технологій для розробки системи містить обґрунтування вибору технологій, порівняння з аналогами, переваги обраних технологій та короткий опис.

Тексти програмного коду містить в собі символічний запис програми на мові програмування, що була обрана в результаті вибору технології програмування.

ПМВ містить опис випробування програми та тестові сценарії для перевірки програмного продукту на відповідність функціональним вимогам.

5 СКЛАД І ПОРЯДОК ВИПРОБУВАНЬ

Тести включають перевірку основної функціональності – навігація сайтом, авторизація та реєстрація, аналіз зображень та збереження результатів. Тестові сценарії та результат їх проходження наведений у таблицях 5.1 – 5.7.

Таблиця 5.1 – Тестування форми авторизації на відкриття сторінки діаграми колеса життя

Тест:	Форма авторизації при успішній авторизації відкриває сторінку діаграми колеса життя
Початковий стан системи:	Відкрита сторінка з формою авторизації
Дія:	Вводимо правильний логін та пароль і натискаємо кнопку «вхід»
Очікуваний результат:	Відкривається сторінка діаграми колеса життя; В меню відображається ім’я поточного користувача
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.2 – Тестування меню користувача – гостя

Тест:	В меню користувача відображені посилання на доступні незареєстрованому користувачу системи
Початковий стан системи:	Відкрита домашня сторінка; користувач не ввійшов в систему
Дія:	-
Очікуваний результат:	Користувач бачить кнопку «почати роботу», посилання на сторінку «авторизація»
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.3 – Тестування меню авторизованого користувача

Тест:	В меню користувача відображені посилання на доступні авторизованому користувачу
Початковий стан системи:	Відкрита сторінка діаграми колеса життя; користувач ввійшов в систему
Дія:	-
Очікуваний результат:	Користувач бачить назву застосунку, посилання на сторінки «Статистика», «Цілі», «Колесо Життя», «Календар» та своє ім’я
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.4 – Тестування меню користувача, посилання на сторінку статистики

Тест:	Посилання на сторінку статистики в меню користувача відкриває сторінку статистики
Початковий стан системи:	Відкрита сторінка діаграми колеса життя; користувач ввійшов в систему
Дія:	Натискаємо на посилання «Statistic»
Очікуваний результат:	Перехід на сторінку статистики
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.5 – Тестування меню користувача, посилання на сторінку заповнення цілей

Тест:	Посилання на сторінку заповнення цілей в меню користувача відкриває сторінку заповнення цілей
Початковий стан системи:	Відкрита сторінка діаграми колеса життя; користувач ввійшов в систему
Дія:	Натискаємо на посилання «Aims»
Очікуваний результат:	Перехід на сторінку заповнення цілей
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.6 – Тестування меню користувача, посилання на календарного плану

Тест:	Посилання на сторінку календарного плану в меню користувача відкриває сторінку календарного плану
Початковий стан системи:	Відкрита сторінка діаграми колеса життя; користувач ввійшов в систему
Дія:	Натискаємо на посилання «Calendar»
Очікуваний результат:	Перехід на сторінку календарного плану
Фактичний результат співпадає з очікуваним:	Так

Таблиця 5.7 – Тестування перестановки розкладу для авторизованого користувача

Тест:	На сторінці календарного плану авторизований користувач має змогу впорядкувати свій розклад
Початковий стан системи:	Відкрита сторінка календарного плану; користувач ввійшов в систему
Дія:	Натискаємо на посилання «Sort Schedule»
Очікуваний результат:	На сторінці відображається новий календарний план вже з відсортованим списком завдань
Фактичний результат співпадає з очікуваним:	Так

6 МЕТОДИ ВИПРОБУВАНЬ

Метод даних випробувань – ручне тестування на інтеграційному рівні. Всі випробування пройдено успішно.

Графічний матеріал
до дипломного проекту

на тему: Інформаційна система підтримки розвитку та самоорганізації особистості

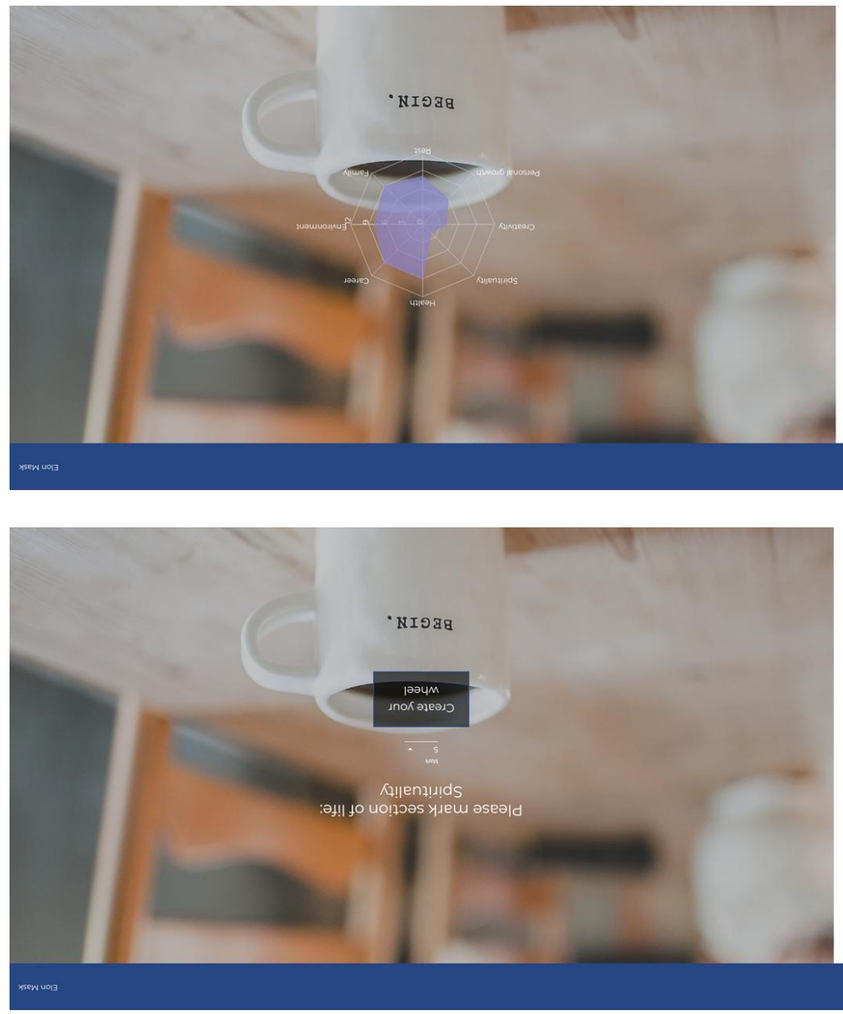
Київ – 2019 року

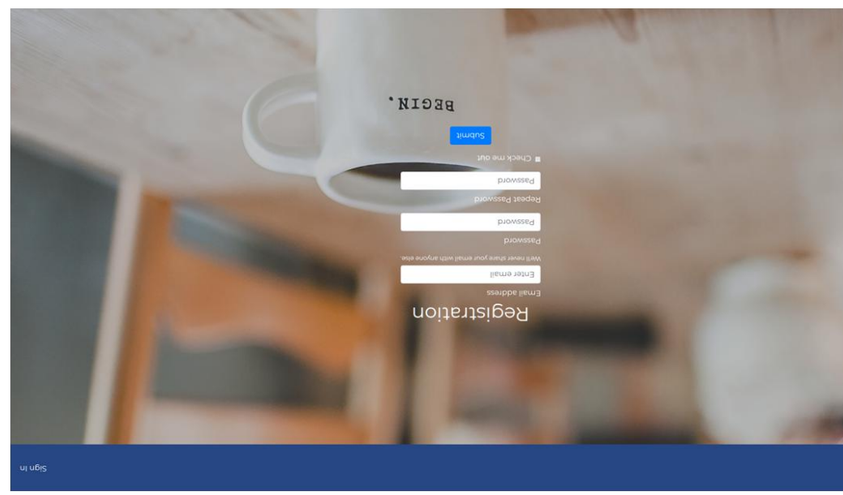
					ДП ІС-5224.1181-с.ССД	
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна діяльності	Літера / Маса / Масштаб
Розробив		Черков П.Ю.				
Перевірив		Сперкач М.О.				Аркуш 1 / Аркушів 3
Т. кон.					Інформаційна система підтримки розвитку та самоорганізації особистості	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52
Н. кон.		Халус О.А.				
Затвердив		Сперкач М.О.				

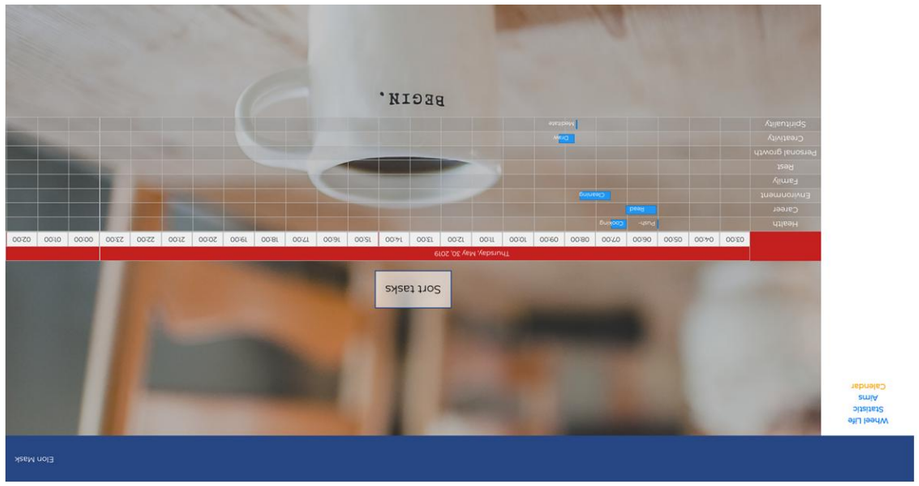
					ДП ІС-5224.1181-с.ССД	
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна діяльності	Літера / Маса / Масштаб
Розробив		Черков П.Ю.				Аркуш 2 / Аркушів 3
Перевірив		Сперкач М.О.				
Т. кон.					Інформаційна система підтримки розвитку та самоорганізації особистості	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52
Н. кон.		Халус О.А.				
Затвердив		Сперкач М.О.				

					ДП ІС-5224.1181-с.ССД		
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна діяльності	Літера	Маса / Масштаб
Розробив		Черков П.Ю.					
Перевірив		Сперкач М.О.				Аркуш 3	Аркушів 3
Т. кон.					Інформаційна система підтримки розвитку та самоорганізації особистості	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52	
Н. кон.		Халус О.А.					
Затвердив		Сперкач М.О.					

					ДП ІС-5224.1181-с.СБД		
Зм.	Арк.	№ документа	Підпис	Дата	Схема бази даних	Літера	Маса / Масштаб
Розробив		Черков П.Ю.					
Перевірив		Сперкач М.О.				Аркуш 1	Аркушів 1
Т. кон.					Інформаційна система підтримки розвитку та самоорганізації особистості	*КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52*	
Н. кон.		Халус О.А.					
Затвердив		Сперкач М.О.					







Зм.	Арк.	№ докумeнта	Пiдпис	Дата	
Розробив		Чернов П.Ю.			Кресленна витягу екранних форм
Перевірив		Снєжин М.О.			
Т. контр.					
Н. контр.		Харус О.А.			Інформаційна система підтримки розвитку та самопрагнень особистості
Затвердив		Снєжин М.О.			

ДП ІС-5224.1181-с.КЕ

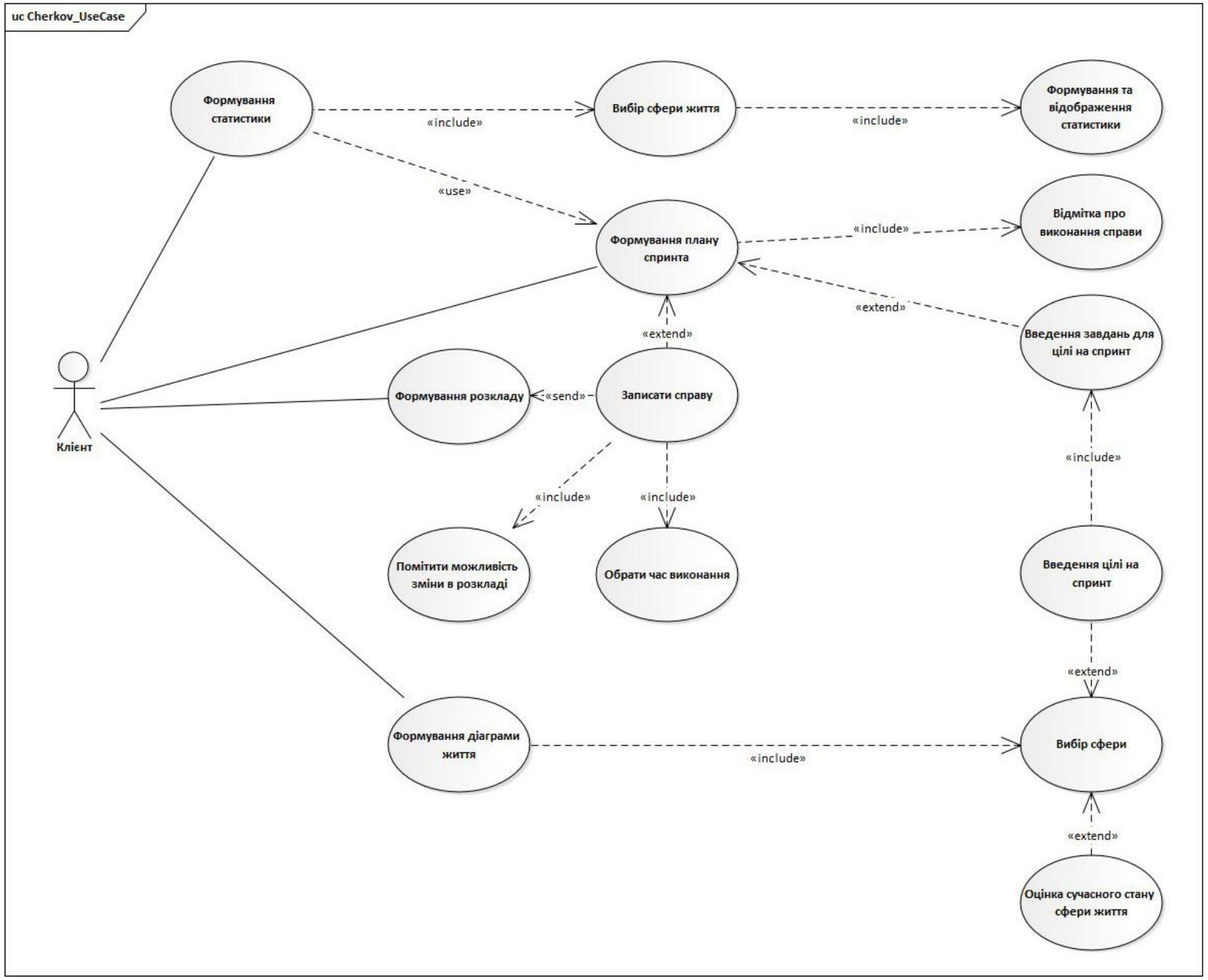
Літера	Маса	Масштаб

Аркуш 1	Аркушів 1

КПІ ім. Ігоря Сікорського
кафедра АСОІУ гр. ІС-52

ДП ІС-5224.1181-с.КЕ

					ДП ІС-5224.1181-с.ССП		
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна послідовності	Літера	Маса / Масштаб
Розробив		Черков П.Ю.					
Перевірив		Сперкач М.О.				Аркуш 1	Аркушів 1
Т. кон.					Інформаційна система підтримки розвитку та самоорганізації особистості	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52	
Н. кон.		Халус О.А.					
Затвердив		Сперкач М.О.					



					ДП ІС-5224.1181-с.ССВ	
						Літера / Маса / Масштаб
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна варіантів використань	
Розробив		Черков П.Ю.				Аркуш 1 / Аркушів 1
Перевірив		Сперкач М.О.				
Т. кон.					Інформаційна система підтримки розвитку та самоорганізації особистості	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52
Н. кон.		Халус О.А.				
Затвердив		Сперкач М.О.				

Змін.	Арк.	№ докумен.	Підпис	Дата	ДП ІС-5224.1181-с.ССК		
Розробив		Черков П.Ю.			Структурна схема компонентів	Літера	Маса Масштаб
Перевірив		Скеркач М.О.				Аркуш 1	Аркушів 1
Т. контр.					Інформаційна система підтримки самоорганізації та розвитку особистості	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52	
Н. контр.		Харус О.А.					
Затвердив		Скеркач М.О.